SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1a. REPORT SECURITY CLASSIFICATION Unclassified	1b. RESTRICTIVE MARKINGS None
2a. SECURITY CLASSIFICATION AUTHORITY N/A	3. DISTRIBUTION / AVAILABILITY OF REPORT Unlimited
2b. DECLASSIFICATION / DOWNGRADING SCHEDULE N/A	
4. PERFORMING ORGANIZATION REPORT NUMBER(S) N/A	5. MONITORING ORGANIZATION REPORT NUMBER(S) N/A
6a. NAME OF PERFORMING ORGANIZATION University of Washington	6b. OFFICE SYMBOL (If applicable)
6c. ADDRESS (City, State, and ZIP Code) Seattle, WA 98195	7a. NAME OF MONITORING ORGANIZATION PERSEREC 7b. ADDRESS (City, State, and ZIP Code) 99 Pacific Street, Building 455-E Monterey, CA 93940
8a. NAME OF FUNDING / SPONSORING ORGANIZATION Office of Naval Research	8b. OFFICE SYMBOL (If applicable)
8c. ADDRESS (City, State, and ZIP Code) Arlington, VA 22217	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER N00014-93-1-0451 10. SOURCE OF FUNDING NUMBERS

PROGRAM ELEMENT NO.	PROJECT NO.	TASK NO.	WORK UNIT ACCESSION NO.

11. TITLE (Include Security Classification)
Improving the Reliability of Polygraph Tests (Unclassified)

12. PERSONAL AUTHOR(S)
Clarkson, Douglas B. and Martin, Doug

13a. TYPE OF REPORT	13b. TIME COVERED	14. DATE OF REPORT (Year, Month, Day)	15. PAGE COUNT
Final	FROM 1/93 TO 1/94	1994 January 27	112

16. SUPPLEMENTARY NOTATION

17. COSATI CODES			18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
FIELD	GROUP	SUB-GROUP	Polygraph, Detection of Deception, Lie Detector

19. ABSTRACT (Continue on reverse if necessary and identify by block number)

The report summarizes the work completed during the second year of a two year study to examine the efficacy of using tree-based classifiers to analyze polygraph charts produced in a detection of deception context. Four hundred twenty-six (426) Modified General Question Technique (MGQT) polygraph charts produced by 144 subjects were used in this phase of the study. Half of the charts were used to develop the tree-based classifiers, the other half were used for cross-validation including estimating the probabilities of misclassification for each of the tree classifiers tested. The lowest misclassification probability achieved was 13.9%. Detailed information is provided about research procedures employed to both capture and analyze the data.

20. DISTRIBUTION / AVAILABILITY OF ABSTRACT ☒ UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT. ☐ DTIC USERS	21. ABSTRACT SECURITY CLASSIFICATION Unclassified
22a. NAME OF RESPONSIBLE INDIVIDUAL Howard W. Timm	22b. TELEPHONE (Include Area Code) 408-656-2448 22c. OFFICE SYMBOL PERSEREC

DD Form 1473, JUN 86 *Previous editions are obsolete.* SECURITY CLASSIFICATION OF THIS PAGE

S/N 0102-LF-014-6603

JCH

19950810 006

Per Anne Watson at Office of Naval research Arlington Va (DSN: 226-4108) they sent what they have and unable to get the missing pages.

October 2, 1995

Polygraph Reliability: Year Two Report

Douglas B. Clarkson and Doug Martin

January 27, 1994

1 Overview

This document summarizes the work on the NSA Polygraph project at the University of Washington from October, 1992 through October, 1993 (year two of a two year study). The reader should be familiar with the year one report on this project which gives a basic outline of the polygraph problem and lists the features (summary statistics) that are used to classify charts at the University of Washington. The year one report is reproduced as Appendix I.

Some disruption in year two progress was caused by the departure of Chris Pounds during June of 1993. After some time delay we were able to engage the services of Douglas B. Clarkson, Ph.D., to replace Pounds for the remainder of the year two work.

We give a summary of the work during the first part of year two (the period prior to Pounds departure) in the first section of this document, the executive summary. The first section also contains a brief summary of the work performed after June 30, 1993 (after the departure of Pounds and the arrival of Clarkson). A more detailed description of the work for the first part of year two can be found in the *PolyGraph Reliability: Year Two Report : Part 1*, reproduced as Appendix II.

Following the executive summary, a more detailed description of the work performed since June, 1993 is given. In the final section we again summarize our findings, and give areas where further study might prove useful.

2 Executive Summary

Polygraph tests are used to determine if a subject is being deceptive in their response to one or more of a series of questions. Two kinds of polygraph tests were analysed during the course of this study. A ZONE of comparison polygraph test is concerned with deception regarding a single question or act. In this test multiple "relevant" questions concerning the same act are asked. Each relevant question is paired with a single "control" question and differences in response between the paired control and relevant questions are observed. In the MGQT (modified general question technique) test a set of four relevant questions in possibly different subject areas are asked along with two control questions. The MGQT test is more of a "fishing" expedition in the sense that questions about more than one event may be asked. Thus, it is possible on the MGQT for the subject to be deceptive on no questions, on a few questions, or on all four questions. This is in marked contrast with the zone of comparison test in which the subject is either deceptive on all questions, or non-deceptive on all questions.

During October and November of 1992, most work involved verifying the integrity and quality of the ZONE data charts in the S-PLUS database. December's work concerned streamlining the functions used to build features in the first year, building these functions around a modern object-oriented paradigm. January and February of 1993 were data management months. The previous year's work was condensed and organized on the computing systems. Work on the MGQT charts began in March. During this month, the files on some charts were converted from Axciton format into S-PLUS format. Work on these initial MGQT charts continued through the middle of June, at which point Clarkson became responsible for the programming work, and Pounds left the University of Washington (and this study) for employment in the private sector. The work performed from October, 1992, through June, 1993, is summarized in Appendix II, written by Pounds and Martin, *Polygraph Reliability: Year Two Report : Part 1.*

After a summer 1993 hiatus, work resumed in of September, 1993. During this month, and much of October, the Polygraph software was reorganized and documented, and an additional 232 MGQT charts were integrated into the database. During this time, all of

the MGQT charts were also screened for abnormalities, and abnormal charts were removed. Only serious abnormalities lead to removal. These included charts in which the wrong types of questions were asked (e.g., charts for "stim" tests), charts in which it was impossible to determine the question order, and charts in which the polygraph recording was clearly erroneous. A listing of the charts removed from the analysis is given in Appendix III. After removing the "bad" charts, a total of 426 charts on 144 subjects were analyzed.

A "feature" is a statistic summarizing some aspect of a polygraph chart. We computed only two types of features: features based upon simple summary statistics of a single channels data, and features based upon a *frequency domain* analysis of one or two channels. In all cases we analyzed sections of the chart following the beginning of a question segment. A detailed listing of the features we used is given in Appendix IV. A general description of how features are computed is given in the *Year One Report*, Appendix I. Variations used in this study are described below.

After the initial data cleaning, a number of preliminary analyses were performed. These were designed to tell us whether or not our feature vectors were being computed correctly. Analyses verifying our algorithm for computing the fundamental frequency of a chart, looking for outliers, and verifying our choice of the span constant in a fixed span smoother were performed. These analyses were basically visual, and were usually perfomed upon only a few charts. The empirical evidence seemed to justify our choices, as is discussed below.

After these preliminary analyses, we began an analysis of the polygraph test data using the features we had computed. A tree-based classifier was used (see Appendix I) to classify each subject as being deceptive or non-deceptive. The methodology used here varied from the methods used in the year one report as follows: In year one charts were first classified as deceptive or non-deceptive using one tree classifier, and the subjects classification was then based upon a second tree built from the deceptive/non-deceptive classification scores obtained from the three charts collected on each subject. We felt that a better classifier would simulataneously consider the information from all three charts, and classify the subjects directly. This was done by combining features over the three charts for a single subject. Alternatively, charts were classified as having or not having anomalies, and the first two

"good" chart (in the sense of few anomalies) were used to obtain a single subject record.

For all features, the feature value was computed as a comparison of the subjects score on a relevant versus a control question. In the ZONE test there were an equal number of control and relevant questions, and the control question immediately preceding the relevant question was used in the comparison. This was not the case in the MGQT test, where typically four relevant but only two control questions were asked. A major consideration in the MGQT test was in deciding how to combine the control and relevant questions. This question was solved here by avoiding a decision: each relevant question on a chart was compared (via differences) with both of the control questions on the chart. In addition, the minimums and the maximums of these eight possible differences were used as an alternate question combining strategy.

A large number of tree classifiers were obtained: each of a total of eight sets of features resulted in four tree classifiers, two for the two chart combining methods, times two for the two ways to compare the relevant and control questions. In addition, we constructed tree classifiers for features purely in the frequency domain, constructed tree classifiers for features considered to be summary statistics, and we constructed tree classifiers based upon the combined set of features. We also used split samples to obtain unbiased estimates of the probabilities of misclassification, and this required additional tree classifiers based upon the split samples.

The results obtained from our tree classifiers indicate that there is substantial information regarding deception in features based upon descriptive statistics and also in features based upon the the frequency domain. However, the frequency domain features seldom entered into trees constructed from the combination of both sets of features, indicating that the frequency domain features contained little additional information regarding deception when compared with features based upon simple summary statistics. Moreover, the frequency domain features tended to result in tree classifiers which were less discriminating than tree classifiers based upon descriptive statistics.

The probabilities of misclassification we were able to achieve were highly variable because of the small number (24) of non-deceptive subjects available. Moreover, the estimated

probabilities of misclassification than we obtained were larger than we would have preferred. For example, when split samples were used in which one-half of the data was used to fit the tree, and the second half of the data was used to estimate probabilities of misclassification, we never achieved a misclassification probability smaller than (13.9) percent in any of our tree classifiers. Notice that this split-sample tree-based classifier analysis was based upon only 72 subjects (and only 12 non-deceptive subjects), not a large number of subjects for a tree based classifier.

We considered using an inclonclusive category, but the nature of our tree-based classifier, and the relative lack of non-deceptive subjects, seemed to rule out this possibility, as is discussed below.

3 Analysis Details

The MGQT (modified general question technique) polygraph test asks four relevant questions in up to four different areas. The subject may be deceptive on no relevant questions, on a few relevant questions, or on all relevant questions. This aspect of MGQT test makes it particularly difficult to study. Because a deceptive subject may be deceptive on as few as one of the relevant questions, we should probably classify questions, rather than subjects, as being deceptive or non-deceptive. This is not possible in the MGQT test, however, because the classification of deception is on a subject by subject basis.

On both the the MGQT and the ZONE test, each subject is asked different questions. On the MGQT test, unlike the zone test, the deceptive subject is not necessarily deceptive on all relevant questions. This fact alone tends to wash out the effects of deception in any classifier: both deceptive and non-deceptive subjects can be non-deceptive on some relevant questions. Obtaining a good classifier in the MGQT test is, then, much more difficult than obtaining a good classifier in the ZONE test.

Listed in a logical order in the remainder of this section is a description of the analyses performed at the University of Washington since June of 1993, along with our results.

3.1 Function Modification and Documentation

Clarkson joined the polygraph project on a full time basis in September. Although Pounds provided substantial educational support to Clarkson prior to leaving the project, the number of S-Plus functions and programs which must be learned in the polygraph project is substantial, so, partly as a learning exercise (and partly to further streamline the process), Clarkson slightly modified and then documented the functions that Pounds had been using.

The major modification involved the program which converts the Axciton DOS format binary files into SUN Unix files. By "dyn.load"ing (dynamically loading) this program into S-Plus to make it an S-Plus function, it was possible to eliminate a time consuming and disk-storage-exhausting step in the process used to convert from Axciton File structures to S-Plus file structures. This saved considerable time and disk space usage (about one-half in both cases) over the previous procedure, which required first a conversion to ASCII, and then a conversion to S-Plus file format. By converting directly to the S-Plus format. the ASCII files were completely eliminated.

Function documentation involved writing S-Plus help files for most S-Plus functions used in the Polygraph Study. These help files saved time in the long run because it made the polygraph S-Plus functions much easier to use. This documentation is reproduced here as Appendix V. Because of this work, there now exists a usable system of S-Plus functions for analyzing polygraph data which other polygraph researchers can use.

3.2 Data Cleaning

Pounds had indicated that the initial set of MGQT data had been cleaned in the sense that each chart had been examined to ensure that the questions were all present and that all required data was available from the Axciton files. In early September additional data arrived with an additional 57 charts. To ease the data cleaning effort for these new charts, Clarkson wrote a number of S-Plus functions for displaying the charts, the questions file, and the order of the event markers on the computer screen. These programs were used to clean these 57 new charts, and were then used to visually inspect the charts that Pounds had looked at.

A large number of problems were found in both sets of charts. The type and order of questions in many charts were inappropriate, for example, because the chart contained the results of a "stim" or ZONE test, not an MGQT test. Movement was easily detectable in many charts (these charts were left in the data set), and on some charts one or more of the channels was flat (again, these charts were left in the data set).

By far the biggest problem with the data concerned the question order. In both the new data, and the data that Pounds had initially screened, the order in which questions were asked on the third chart was not identical to the question order used in the first two charts. Repetition of the two control questions on the third chart was also a problem. Discussion with Jeff Johnson at NSA indicated that in the standard MGQT test, the examiner is free to give the questions in any desired order on the final chart. Typically (but not always), the same examiner will permute the final chart questions in some manner. In the charts Clarkson examined, the order of the questions on the first two charts was usually

```
TB I1 I2 R1 I3 R2 C1 I4 R3 R4 C2
```

(on some charts R4 and C2 were interchanged). This is the standard MGQT format. The question order on the last chart was generally

```
TB I4 I1 R2 C1 R1 C2 R4 C1 R3 C2
```

Here `TB` = test begin, `I1`, `I2`, `I3`, and `I4` are irrelevent questions, `C1` and `C2` are control questions, and `R1`, `R2`, `R3`, and `R4` are relevent questions.

The repetition of the final two control questions significantly complicates the analysis as now four control questions are possible (instead of the usual two). Only the first two control questions were used in all of our analyses.

For many subjects the first relevant question was a "sacrifice relevant" question. Sacrifice relevant questions are not normally scored by a polygraph examiner. In this study sacrifice relevant and relevant questions were treated identically.

Around October 15 an additional 175 charts became available. Because of the earlier work at documenting the S-Plus functions and creating functions for cleaning the data, cleaning these 175 charts proceeded quickly. When all of the data was clean, a total of 426 charts

on 144 subjects were available. Of the 426 charts, 354 (or 83.1 percent) were classified as deceptive. Of the 144 subjects, 120 (or 83.3 percent) were deceptive.

3.3 Estimation of the Fundamental Frequencies

The fundamental frequency for the cardiograph or pneumograph channels is the frequency with the greatest energy. Pounds had estimated the fundamental frequency of the cardiograph and pneumograph channels using the S-Plus spec.pgram() function. A problem encountered in this estimation was that, because of heart-lung interactions, the fundamental frequency for the cardiograph channel would occassionaly be estimated at a much too low frequency – the effect of respiration on blood pressure would occasionally be so large that the cardiograph frequency with greatest energy was the frequency of respiration.

The solution Pounds proposed was to require that the cardiograph fundamental frequency be greater than the equivalent of 35.4 heartbeats per minute. He also required that the fundamental frequency on the pneumograph be greater than the equivalent of 3.9 breaths per minute. Plots of the estimated fundamental frequency for the cardiograph versus the estimated fundamental frequency for the pneumograph indicated that Pounds's solution was working correctly. An example is displayed in Figure 1.

3.4 Exploratory Data Analysis

To get a better feel for the data, and to check for anomalies, histograms and quantiles (minimum, lower quartile, median, upper quartile, maximum) were computed for each of the extracted features. Because we expect the distribution of some features to be different for the deceptive versus the non-deceptive subjects, histograms of both the deceptive and the non-deceptive subjects should reflect a mixture distribution.

Anomalies, in the form of obviously large data points, were observed in the means, ranges, and variances of the observed data in the galvanic skin response channel. These problems could be traced to the galvanic skin response channel on the first chart for subject "z8qmts.1". Evidently, the computed features accurately reflect the data — the galvanic skin response channel for this subject is clearly an anomaly and it is possible that the chart should be

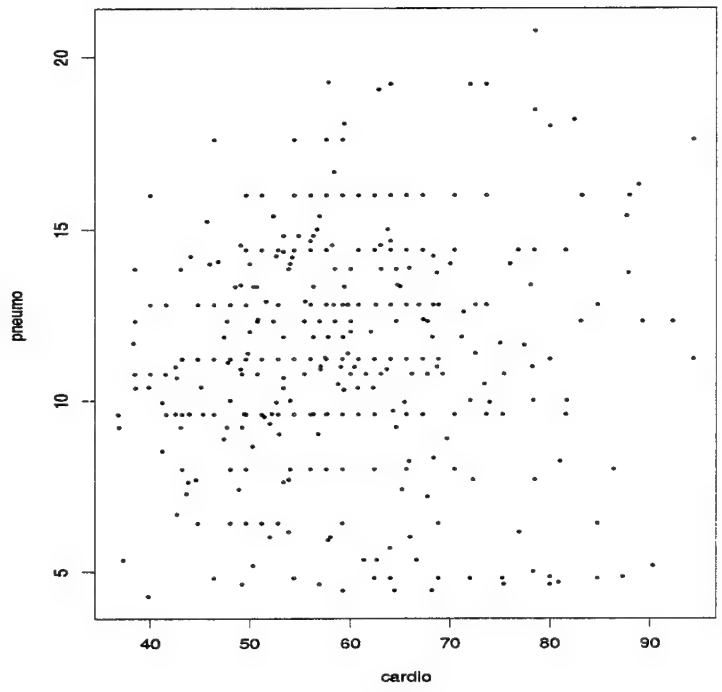

Figure 1: Cardiograph and Pneumograph Fundamental Frequencies

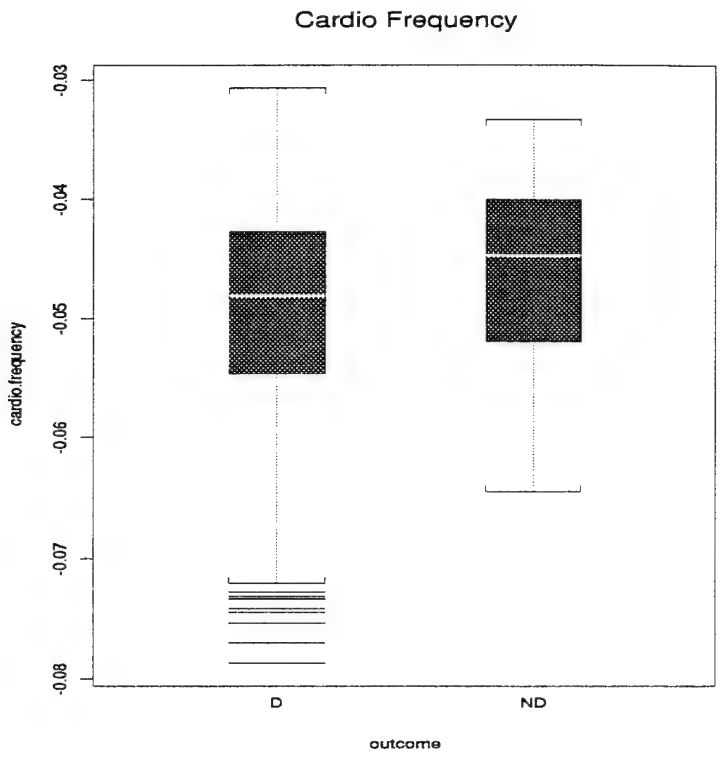

Figure 2: Boxplot of the amplitude of the Fundamental CardioGraph Frequency

removed (it was not). On two charts a few spectral features were computed as NA's (not-a-number), indicating extreme scaling problems in the spectral analysis of the chart data. This missing data was replaced with data from another chart on the same subject.

Boxplots of each feature were computed for the deceptive versus the non-deceptive subjects. These plots invariable indicated small but likely differences between the distributions for the two groups. An example boxplot is given in Figure 2.

3.5 Standardization

The scale of polygraph data collected via the Axciton polygraph varies from chart to chart as the polygraph examiner adjusts the sensitivity of the instrument to account for subject variation. The pattern of variation in the chart data, not its magnitude or scale, is important. For a tree based classifier to make sense, the polygraph data must be standardized to a common location and scale. Pounds and Martin had previously arrived at a standardization

scheme (discussed below) which seemed to work well, and no additional work on standardization methods was performed in the June through November, 1993, time frame. Notice, however, that the smoothing discussed in the next section can also be considered as a form of standardization.

Because of variations in level over the time frame of the chart, standardization was performed at the question level rather than at the chart level. Rather than standardizing the control and relevant questions seperately (and thus losing potential location and scale differences), standardization of the relevant question and the appropriate control question pair was performed simultaneously as follows: The time series associated with each pair of control and relevant questions on each channel were combined to obtain a single series. Standardization of this series then consisted of subtracting the median series value from each element in the series, and dividing each element in the series by the mean absolute deviation of the series.

3.6 Fixed Span Smoother

Over the course of a test the instruments may record at varying levels because of instrument adjustments and/or subject movements. A fixed span smoother (here, super smoother, Friedman, 1984) can remove much of this variation, which we assumed to be of no relevance in the polygraph test. By removing these long term trends attributable to instrument variations, we hoped to be able to better compare control question versus relevant question features. This type of smoothing was used in some of the tree-based classifiers, and was not used in others.

In the MGQT test, unlike the ZONE test, the control and relevant questions are not necessarily adjacent. Rather, the two questions can be three or more questions apart. By removing the charts long term trend we hoped to be able to move the time series associated with the control and relevant questions to the same central location prior to standardizing the data and computing the features. Once the smooth is computed, trend is removed by subtracting the smooth from the raw data. The features are computed on the resulting series. This kind of smoothing was perfromed over the entire channel on each chart.

The "span" in super smoother is the fraction of observations in the smoothing window. When removing trend, what is needed is a span which removes trend due to instrument variations, but which leaves intact local variation due to subject response. If the span is too large, then all variation is ignored, and the smooth becomes a measure of central value. On the other hand, if the span is too small, then the smooth will reflect local variation due to subject response as well as long term trend due to instrumentation changes, and both will subsequently removed from the charts data. Then a span which is large enough to eliminate local variation but not so large as to eliminate all variation is required. Also notice that a small span lessens the effects of observations far removed from the time point under consideration. This is important if, say, the examiner adjusts a channel during an irrelevant question. We do not want the adjustment to be reflected in a relevant or control question. For these reasons, we want the smallest span possible that does not also remove local variation due to a subjects response.

A graphical analysis was used to determine the span for the smooth. Before discussing the results of this analysis, notice that since a chart contains, on average, roughly ten times as much information as is available in a single question, we would expect the correct "span" on a chart to be roughly one tenth the correct span to use on a single question.

Pounds had estimated the best span (again using graphical methods) when using two-questions, a control and a relevant question. He found a span of 0.6 did a good job of maintaining local variation and removing long term trend. Our work verified this result. We also found that a span of 0.06 worked well when smoothing the entire chart. This is consistent with the span of 0.6 when smoothing each question separately. An example smooth for the lower pneumograph channel is given in Figure 3. In this figure the top series is raw data, the middle series is the estimated smooth, and the lower series is the raw data minus its smooth.

Smooths were used for two additional purposes: The spectral estimation proecedure works best if long term trends are removed prior to estimating the spectrum. Also, especially with respect to the cardio channel, it is often easier to see important features or trends in the data when a smooth is used in place of the raw data. The idea is to look more at blood pressure, and less at heart beats. One set of feature vectors used a smooth of the data when

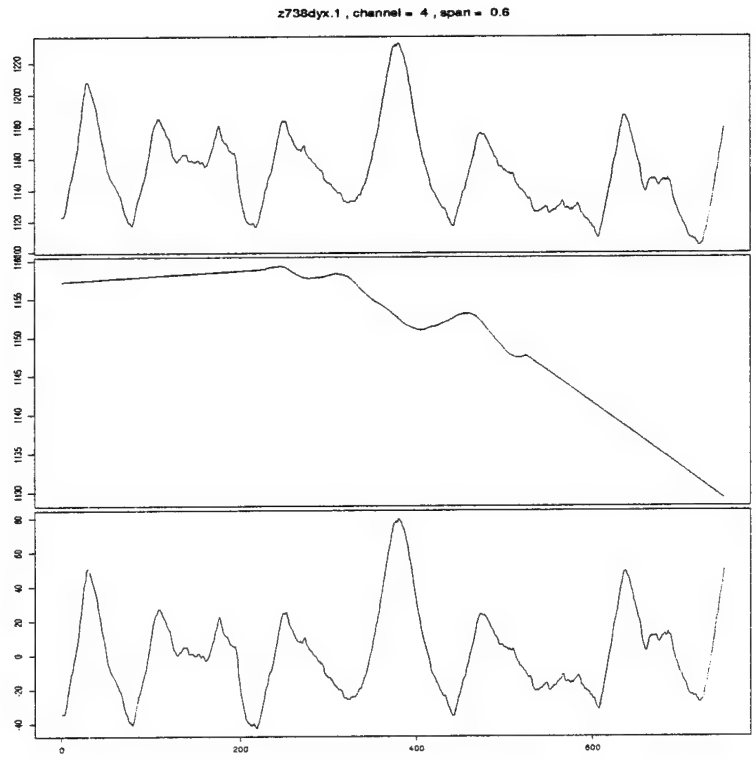

Figure 3: Smoothing an Entire PneumoGraph Chart

computing some simple summary statistics.

3.7 Evaluating the Charts

Anomalies were visible on many charts. Some anomalies led us to simply remove the chart from further consideration: the question order was unknown or wrong, a channel was unusable, the event markers were not present, or if present, were in the wrong order, etc. Other anomalies such as jumps in one or more channels, a flat galvanic skin response for some or all of the data, or an unusual galvanic skin response (very low readings in an irrelevant question, etc.) could not be handled so easily. Charts with these anomalies were categorized from one to three, with one being a "good" chart with no anomalies, and two and three being progressively worse. In some analyses the charts with scores of three were removed. The hope was that with these charts removed, we would obtain better tree classifiers.

If the process of scoring charts were to be repeated and the charts reclassiffied, slightly different results would probably be obtained. Even so, it is unlikely that a chart we rated a "1" would be reclassified as a "3", and vice versa. Of 426 charts, 280 were rated good and scored as a "1" (132 subjects), 117 were rated worse and scored as a "2" (84 subjects), and 29 were rated "bad" and scored as a "3" (26 subjects). Thus, 26/426 or about 7 percent of the charts were removed in some analyses.

3.8 New Galvanic Skin Response Features

It was felt that distance measures between the spectra of the galvanic skin response channel might be useful for classifying deceptive versus non-deceptive individuals. These measures consist of the maximum absolute difference between the integrated spectrum for the relevant versus the control questions, the sum of these absolute differences, and the sum of the squares of these differences. These spectral statistics were computed for smoothed (and standardized) charts (using a smoothing span of 0.6), and for standardized charts without smoothing. By "smoothed charts" we mean the smooth of the time series, not the difference between the input data and the smooth. When smoothed data was used, the smoothing was performed on a question by question basis using a span of 0.6. As it turned out, these new features did

not seem to help much in the tree classifier.

3.9 New Cardiograph and Pneumograph Features

Comparison of our features with those computed by APL revealed a number of possible new features. Among these were features based upon the difference of two smooths on a channel. The first smooth had a narrow span (span = 0.02) in order to capture local structure, while the second smooth had a wider span (span = 0.06) in order to eliminate global trend. The final channel data was taken as the difference between the two smooths. Simple summary statistics (minimums, maximums, and midranges) were computed as features. For some features, a lag one difference was used. The smoothed features were often encountered in the optimal tree classifier.

3.10 Pairing of Control and Relevant Questions

In the zone tests each relevant question was paired with the closest control question. This was also possible in the MGQT tests, but rather than pairing with the closest control question, it was decided to pair with each of the control questions. Because the order and number of control questions was often different on the third chart, only the first occurence of each control questions was used (on most charts each control question was only asked once.) This type of pairing doubled the number of features to be used in constructing classification trees.

3.11 Chart Combining Methods

Previous work at the University of Washington has produced tree classifiers which predicted deception on each chart. These chart classifications were then combined over all charts on a subject using a second tree classifier based upon the chart classification scores. This second classifier was used to classify each subject. Because we felt that information was lost in this two stage procedure, the classification in year two was based upon a single tree classifier which could be used to classify subjects directly. This required a specification of methods for combining the charts on a subject to obtain a single subject feature vector. Two such

methods were used. In the first, all three charts (actually, the first three valid charts) on a subject were combined into a single subject record (with three times as many features). If only two charts were available, the second chart was duplicated as the third chart so that all subjects had complete data records.

The second method for combining subject charts attempted to select only charts which were relatively free from anomalies. As discussed above, some charts were rated as better than others with respect to the presence or absence of anomalies. Using these scores, we combined the first two charts on a subject which we had scored as a "1" or as a "2". In this second method, if only one "good" chart was available on a subject, this chart would be duplicated and used as the final chart for the subject.

3.12 Computing a Tree Classifier

At this point it is worth reviewing the steps involved in computing a tree classifier. After the raw data is moved into the S-plus database, eight matrices summarizing this data are computed. These matrices contain the feature vectors to be used in constructing the tree classifiers. There is one row in each matrix for each chart in the database.

A feature is a statistic computed using a control and a relevant question on a channel of a chart. Each combination of the four relevant questions is combined with each of the two control questions, usually as the difference between the relevant and control questions. Each chart has eight such combinations. Most features are also computed for each of the three channels on the chart (galvanic skin response, cardiograph, and lower pneumograph). Then there are 24 occurances of the feature statistic on each chart. The tree classifier is constructed using as many as three charts per subject. Then in the tree classifier, a feature statistic can appear 72 times for each subject. Typically there are six features per feature set, so that the tree classifier is often based upon 6*72 or 432 feature variables on each subject. Notice that, at best, there are at most 144 subjects.

There are eight sets of feature statistics. Four of these sets are based upon summary statistic features, and four sets are based upon the frequency domain. Variables from each of these two groupings of four feature sets were selected as follows: Each set was input into

the do.tree() function in S-Plus. The do.tree() function merged the charts on each subject to obtain 144 records, one for each of the subjects. Two methods of merging the charts were used, as is discussed above. (In the first chart combining method, the first three charts for a subject are concatenated, in order, to obtain the single subject record. If only two charts were available, the second chart was duplicated and its features were used as the features in the third chart. In the second chart combining method, only two charts were combined, but charts which were rated "3" with respect to chart anomalies, the worst charts, were removed from the database. If only one good chart was available on a subject, its values are duplicated for the second chart.) In both chart combining methods it was assumed that the charts follow a time sequence, with chart one being the first chart recorded, chart two being the second chart recorded, etc.

Once the charts had been combined, a single tree was computed for the combined charts. Standard practice with classification trees requires that the initial classification tree contain more nodes than are really required – the tree is overfit with too many nodes. Cross-validation is then used with this overfit tree to determine the number of nodes which should be used in the final tree classifier. After performing this cross-validation, the do.tree() function would save all of the features vectors from the initial overfit tree for later use.

3.12.1 Preliminary Variable Selection

As indicated above, the study design required the examination of a large number of features (58 in all). This number of features is then multiplied by as much as 72 to obtain the final variables to be used in constructing a tree classifier. Since there are only 144 subjects a variable selection procedure was required to obtain the most important features from the 58 times 72 possible features.

The variable selection strategy used is as follows: The eight sets of features were split into two distinct groups – features involving simple summary statistics, and frequency domain features. Label the sets of features based upon simple summary statistics as `GS1`, `G1`, `G2`, and `G2A`, and label the frequency domain features as `G3`, `G4`, `G5`, and `FD2`. For each matrix in each set, the do.tree() function is used to create a tree, and the variables in the initial

overfit tree classifier are saved in a data frame. Two such data frames were obtained, one for the simple summary statistics, and a second for the frequency domain features. These two data frames are then combined into a third data frame containing both sets of features. The final trees, one for the simple summary statistics, one for the frequency domain variables, and one for all features combined, are computed from these data frames.

3.13 The Tree Classifiers

Rather than give all computed tree classifiers, information on only twelve will be given. All variables selected in the overfit tree are given. A "*" is used to indicated variables that were also included in the final cross-validated tree classifier containing the number of nodes indicated by the cross-validation. The channel numbers are as follows: 1 - GSR, 2 - cardio, and 4 - pneumo. The control and relevant question used in computing the feature are given, as is the chart from which the feature originates. Variable names are listed in Appendix IV with a brief description.

- Chart combining yields three charts per subject
  - Question comparing looks at all possible control–relevant question combinations
    - **Model 1** – Simple summary statistics
      1. *mid.9.1, channel 1, R4 - C1, chart 2
      2. *midrange, channel 1, R3 - C1, chart 3
      3. SP25, channel 1, R4 - C2, chart 2
      4. SP75, channel 1, R2 - C2, chart 1
      5. *maximum, channel 2, R1 - C2, channel 2
      6. midrange, channel 4, R2 - C1, chart 2
    - **Model 2** – Frequency domain
      1. *phase1, channels 2 and 4, R1 - C2, chart 3
      2. *amp1h2, channels 2 and 4, R2 - C1, chart 3
      3. amp1h1, channels 2 and 4, R1 - C2, chart 1

4. coh1h3, channels 2 and 4, R4 - C2, chart 3
5. mxabsm, channel 1, R1-C2, chart 2
6. coherr2, channels 2 and 4, R2 - C2, chart 1
7. fundfreq, channel2, R1 - C2, chart 1

* Model 3 – Frequency domain and simple summary statistics
  1. *mid.9.1, channel 1, R4 - C1, chart 2
  2. *midrange, channel 1, R3 - C1, chart 3
  3. SP25, channel 1, R4 - C2, chart 2
  4. SP75, channel 1, R2 - C2, chart 1
  5. *maximum, channel 2, R1 - C2, channel 2
  6. *midrange, channel 4, R2 - C1, chart 2

– Question comparing looks at minimum and maximums of all possible control–relevant combinations

* **Model 4** – Simple summary statistics
  1. *MAXAMP, channel 1, max, chart 2
  2. *SP75, channel 4, min, chart 2
  3. med240, channel 1, max, chart 2
  4. variance, channel 4, min, chart 1
  5. *midi.8.2, channel 1, min, chart1
  6. *midi.8.2, channel4, max, chart 1
  7. maximum, channel 1, max, chart 1
  8. med240, channel 2, max, chart 1

* **Model 5** – Frequency domain
  1. *smsqsm, channel 1, max, chart 1
  2. *amp1h2, channels 2 and 4, min, chart 1
  3. fund2, channels 2 and 4, max, chart 2
  4. phase1, channels 2 and 4, min, chart 2

5. phaser1, channels 2 and 4, max, chart 2
6. coh1h1, channels 2 and 4, min, chart 2
7. coher2, channels 2 and 4, max, chart 1

* Model 6 – Frequency domain and simple summary statistics
  1. *MAXAMP, channel 1, max, chart 2
  2. *mxabrw, channel 2, max, chart 1
  3. SP75, channel 4, min, chart 2
  4. *minimum, channel 1, max, chart 1
  5. *midi.8.2, channel 1, min, chart 1
  6. midi.8.2, channel4, max, chart 1
  7. fundfreq, channel 2, max, chart 3
  8. range, channel 2, min, chart 1

- Chart combining yields two charts per subject
  - Question comparing looks at all possible control–relevant combinations
    * **Model 7** – Simple summary statistics
      1. *mid.9.1, channel 1, R3 - C2, chart 2
      2. midrange, channel 1, R2 - C1, chart 2
      3. maximum, channel 2, R2 - C1, chart 1
      4. range, channel 1, R3 - C2, chart 2
      5. *median150, channel 2, R3 - C1, chart 2
      6. median60, channel 2, R3 - C1, chart 2
      7. mid.9.1, channel 1, R3 - C1, chart 1
    * **Model 8** – Frequency domain
      1. *mxabsm, channel 1, R1 - C2, chart 1
      2. *amp2a1, channels 2 and 4, R3 - C2, chart 1
      3. *amp1h2, channels 2 and 4, R3 - C2, chart 1

4. choher2, channels 2 and 4, R1 - C1, chart 2
5. coh2, channels 2 and 4, R2 - C2, chart 2
6. coherr1, channels 2 and 4, R1 - C1, chart 1
7. phase 1, channels 2 and 4, R3 - C1, chart 2
8. choher2, channels 2 and 4, R1 - C1, chart 1

* Model 9 – Frequency domain and simple summary statistics
  1. *mid.9.1, channel 1, R3 - C2, chart 2
  2. *midrange, channel 1, R2 - C1, chart 2
  3. maximum, channel 2, R2 - C1, chart 1
  4. range, channel 1, R3 - C2, chart 2
  5. *median150, channel 2, R3 - C1, chart 2
  6. median60, channel 2, R3 - C1, chart 2
  7. fundfreq, channel 2, R1 - C1, chart 2

– Question comparing looks at minimum and maximum of all possible control–relevant combinations

* **Model 10** – Simple summary statistics
  1. *midrange, channel 1, min, chart 1
  2. midrange, channel 4, min, chart 2
  3. midrange, channel 2, min, chart 1
  4. MAXAMP, channel 1, min, chart 2
  5. *minimum, channel 1, min, chart 2
  6. *minimum, channel 1, min, chart 1

* **Model 11** – Frequency domain
  1. *smsqsm, channel 1, max, chart 1
  2. mxabsm, channel 2, max, chart 1
  3. smsqrm, channel 4, min, chart 2
  4. smabrw, channel 4, min, chart 2

5. fund2, channels 2 and 4, max, chart 1
6. coherr2, channels 2 and 4, max, chart 2
7. fundfreq, channel 2, min, chart 2
8. smabrw, channel 2, min, chart 2

* Model 12 – Frequency domain and simple summary statistics
  1. *midrange, channel 1, min, chart 1
  2. midrange, channel 4, min, chart 2
  3. midrange, channel 2, min, chart 1
  4. MAXAMP, channel 1, min, chart 2
  5. *minimum, channel 1, min, chart 2
  6. *minimum, channel 1, min, chart 1

Table 1 summarizes the above models. In this table two estimates of the misclassification probabilities are used. The first two misclassification probabilities with column labels **best** and **indict** are obtained using resampling. They are probably too optimistic. The misclassification probabilities with column labels **split-ND**, **split-D**, and **split** are obtained using split samples. These probabilities are unbiased. Notice that in the split sample technique only 12 non-deceptive subjects were available, so the variability of the estimated probabilities of misclassification is high.

Column labels in the table are as follows:

- **nodes** – the number of nodes indicated by the cross-validation
- **best** – the best estimated probability of misclassificiation obtained using any number of nodes from the overfit tree
- **indict** – the estimated probability of misclassification obtained when a tree with the number of nodes indicated by cross-validation is used
- **split-ND** – the estimated probability of misclassifying a non-deceptive subject when a split sample is used

Table 1: Summary of the Classification Trees

MODEL	nodes	best	indict	split-ND	split-D	split
model 1-3-8-S	4	4/144	12/144	4/12	9/60	13/72
model 2-3-8-F	3	4/144	19/144	8/12	14/60	22/72
model 3-3-8-C	5	4/144	7/144	2/12	12/60	14/72
model 4-3-2-S	5	8/144	9/144	7/12	3/60	10/72
model 5-3-2-F	3	9/144	24/144	11/12	12/60	23/72
model 6-3-2-C	6	7/144	9/144	7/12	3/60	10/72
model 7-2-8-S	3	6/144	17/144	5/12	6/60	11/72
model 8-2-8-F	4	8/144	18/144	8/12	14/60	22/72
model 9-2-8-C	4	6/144	9/144	5/12	6/60	11/72
model 10-2-2-S	4	4/144	12/144	8/12	6/60	14/72
model 11-2-2-F	2	10/144	24/144	11/12	4/60	15/60
model 12-2-2-C	4	4/144	12/144	8/12	6/60	14/60

- **split-D** – the estimated probability of misclassifying a deceptive subject when a split sample is used
- **split** – the estimated probability of misclassification for all subjects when a split sample is used

Following the model label is the number of charts used in the model, either 3 charts, or the first 2 good charts. Following the number of charts used is a number indicating the method used for combining control and relenvant questions. This number is 8 if all combinations of relevant and control questions were used, or 2 if the minimums and maximums were used. Finally, the last model descriptor in the model label is an S for summary statistic features, an F for frequency domain features, or a C for combined summary statistic and frequency domain features.

From the table it is clear that the summary statistic features out perform the frequency domain features. Notice, however, that when combined feature sets are included for possible

selection by the tree classifier, then the frequency domain features can still be important.

The estimated probabilities of misclassification are disappointing. One reason for this may be the paucity of non-deceptive subjects, especially when the split sample is used. While the misclassification probabilities listed under column **indict** seem good, because they are based upon resampling, they may be optimistic.

3.14 An Inconclusive Category

While an inconclusive category could be easily incorporated into the probability estimates, there seems little point in doing so. With only twelve non-deceptive subjects available for classification in the split-sample technique and 24 non-deceptive subjects overall, yet another category of probabilities to estimate would only serve to add variability to the estimates. Moreover, preliminary investigations indicated little benefit from an inconclusive category. The probability of correct classification is already disappointingly low. Use of an inconclusive category would only serve to make it even lower.

4 Summary

A number of problems with the MGQT data have been discussed. Perhaps the biggest problem is the lack of data for non-deceptive subjects. Less than 17 percent of our subjects are non-deceptive. Simply by classifying all subjects as deceptive, an overall error rate of 17 percent can be obtained. Another problem with the small number of non-deceptives subjects occurs in the tree-classifier. Very few non-deceptive nodes can be obtained when only 24 non-deceptive subjects.

Much as we would like to be, it is difficult to be definitive with respect to the information in the frequency domain when classifying subjects in a polygraph test. The frequency domain features are clearly less important than the simple summary statistic features, and yet they appear to have some ability to classify subjects as to their deceptive/non-deceptive status. Because the data is quite messy and because there are relatively few non-deceptive subjects, it does not seem unreasonable to believe that many of the results obtained here are random:

given another set of similar data, quite different variables may enter the model, and we might obtain quite different estimates for our probabilities of misclassification.

1 Appendix I

The year one report.

Year One Report

for

Polygraph Reliability
MDA904-89-C-4216

Contracted through

National Security Agency
Fort Meade, Maryland 20755

Research performed by

R. Douglas Martin, Ph. D.
Christopher B. Pounds

through

The Department of Statistics
Unverisity of Washington
Seattle, Washington 98195

Period: October 1, 1991 – September 30, 1992

CLASSIFICATION TREES AND SPECTRAL ANALYSIS METHODS APPLIED TO POLYGRAPH DATA

R. Douglas Martin, Ph. D. and Christopher Pounds
University of Washington, Seattle

Summary

We have introduced frequency domain spectral analysis techniques for he problem of classifying digitized polygraph data from the Axciton Polygraph System. Our primary classification tool has been the tree based classifier which splits the data into distinct subgroups according to the binary response of "deceptive" or "no deception." The raw signal is standardized by taking control-relevant question pairs as a single observation and robustly standardizing them. Range based features, 1st difference based features and smoothed features are used along with several spectral features to train the classification trees. Cross-validation is used to select a tree based model for each chart and then for each person. A final error rate of 12.6% was obtained for a sample of 199 subjects with two or more charts each.

1 INITIAL WORK

We began background study of this problem in January of 1992, and began working with the Zone Comparison test data in March of 1992, when Chris Pounds visited to Johns Hopkins Applied Physics Laboratory in Maryland. There he met with Dale Olsen and John Harris to discuss data formatting issues. APL's work had been done on the PC platform, and so a considerable time was required to translate the programs to read the digitized data in a form suitable for the UNIX workstation environment at the University of Washington.

APL Data Organization

In June, Chris Pounds returned to APL to present the tree based classification methods to the polygraph examiners. This trip also allowed us to obtain 31 disks containing data for 605 polygraph charts with 219 different names. Because of test interruptions, these 219 names consist of charts from fewer than 219 individuals. APL had organized the data according to time of arrival and source of the data. The "A" dataset consisted of 129 charts received before May of 1992, and the "B" dataset consisted of 90 charts received in May and June of 1992. Within these groupings, the following source groups were separated into different directories: FBI , Clayton County, Vermont State Police, SLED, DEA, Marion County, Birmingham and Anniston. See Figure 1 for an explanation of these steps on the A dataset. A compression algorithm called **pkzip** (which can be used on both PC and UNIX platforms) was required to compact the collection of data onto 31 disks. Each disk represented a subset from each of the source directories.

The Zone-Comparison Test orders questions on a chart in the following format:

1. TB Test is about to begin
2. N Neutral
3. SR Sacrifice Relevant
4. S Symptomatic
5. C1 Control
6. R1 Relevant
7. C2 Control
8. R2 Relevant
9. Sy Symptomatic
10. C3 Control
11. R3 Relevant
12. XX End of Test

Responses are recorded from the following channels:

Galvanic Skin Response
Cardio
Upper Respiratory
Lower Respiratory

The Axciton Polygraph System creates three files for each chart that is processed. Charts from the same subject are labeled with the same prefix and filename extensions depend on the contents and the chart. The "x" below indicates the chart number (ranging from 1 to 5). The "0" is a place holder to give a 3 place extension to the file.

Extension	Contents
0x1	Question and Event Markers
0x2	Chart Data (Compressed)
0x3	Question Text

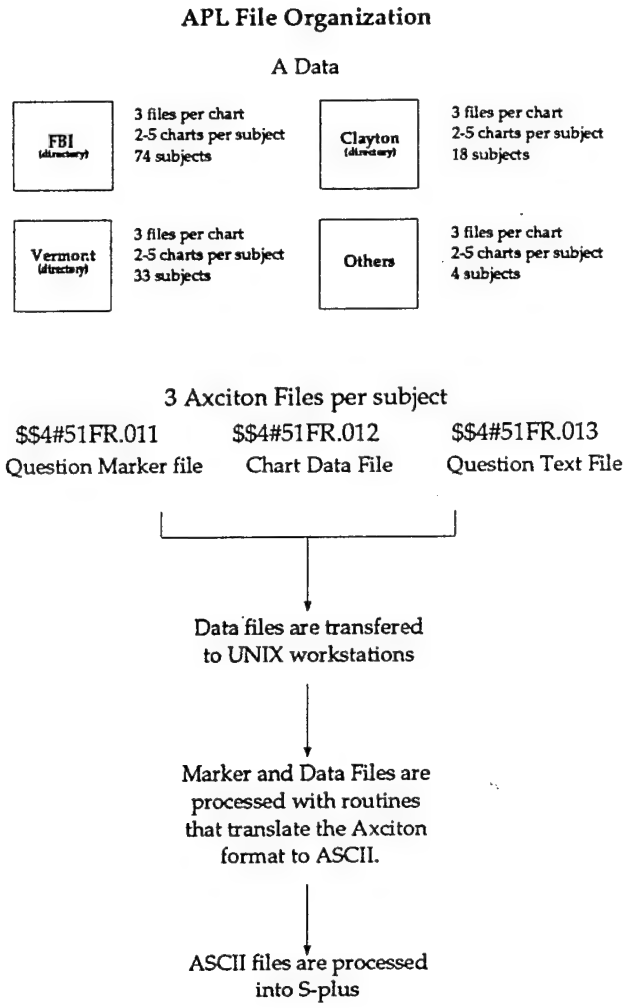

Figure 1: **File Processing Steps**

When the polygraph examiner uses the Axciton System, he presses keys at the beginning and end of each question. Since the questions are given sequentially as listed above, the key presses can be used to label the data with question labels.

2 DATA ORGANIZATION

When dealing with a large database, the system must be organized in such a way to allow the analyst to process and explore the information easily. The S-Plus language is itself very powerful and contains a very extensive analytic toolbox which lends itself to in depth exploration of time series objects. However the language is not particularly powerful as a

database tool. Thus much time and effort at the early part of the contract was spent setting up an efficient polygraph database in S-Plus.

By modifying a program from the APL DOS system, the Axciton files were translated into ASCII and then were arranged into subgroups roughly according to the directories that APL had stored them in. Within the subgroups, they were then processed into S-Plus and the original ASCII files were stored on tape.

S-Plus processing consists of translating the chart data into the matrix data objects used by the S-Plus data analysis system. In addition, the event marker files are processed into separate objects which list the beginning and end of each question and label each question based on the order above. See Figure 2.

After the labeling was completed, the labels were checked for internal consistency. If files followed an abnormal pattern (for example, an uneven number of controls compared to relevants, or fewer than 8 segments) then the charts were examined for movements and artifacts and the question scripts were examined. Some deletions were made at this stage, but further screening is possible.

Once the database and question labels were completed, functions for display and analysis of the database were built. The functions we wrote had four principal arguments to contend with:

1. Which chart or subject should be displayed?
2. What questions within the chart should be shown?
3. Which channels are important?
4. Should any analysis plots be shown? (including detrending, first differences, or spectral representations)

To build up the vector of information from all of the charts, a function loops over each subject and then over the selected subject's charts. For each chart, the channel and questions of interest are selected, and then the feature is calculated for each question of interest. Depending on the complexity of the feature, it took up to five and one-half hours on a Sparc IPC to extract a single feature from the 567 charts. The process was completely automated so the bulk of computing could be accomplished in overnight jobs. These feature vectors were given to the classification tree tools for analysis.

Splus File Organization

FBI 1

FBI 2

FBI 3

Clayton 1

o
o
o

DEA 1

Files are grouped into 11 separate directories for ease of handling.

For each Chart Data file, the Marker file is processed into Splus to make an Splus object which contains character labels (like S, C1, R1, C2) which can be used when accessing the charts. File names are modified to remove symbols (#,@, %) since they are reserved in UNIX. These steps are done on each directory one at a time.

The separate directories are linked with an `attach()` command which allows access to all of charts from a single directory.

It is in this working directory that functions process the charts and the features are stored.

Label object example

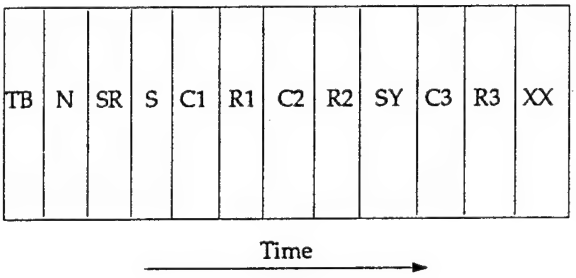

Figure 2: S-Plus Database Organization

An example of a chart plot produced by one of the functions created is given in Figure 3. The first row is the GSR channel; the second row is Cardio. The third row is lower respiratory. Typically the upper and lower respiratory channels are very similar, so the upper channel is often omitted from plots.

3 FEATURE VECTORS

As usual in classification, the tree based method uses *feature vectors* as inputs. We call the coordinates of a feature vector *features*. Because of calibration differences in the the Axciton Polygraph System, tests administered to different subjects will often result in entirely different ranges for the data in any given channel. This is seen in the first row of Figure 4 where the GSR channels have different ranges.

Standardization

Polygraph data is highly variable, due to both instrument calibration variability and cross-individual variability. See for example the raw polygraph GSR channel data for a control-relevant question pair in the first row of Figure 4. Thus it is very important to standardize the raw polygraph data. Initially we tried standardizing each question separately by subtracting off the mean of the data and dividing by the standard deviation. Unfortunately, this method loses useful information contained in the comparative differences between questions.

Instead, we combined the C-R pairs and standardized the entire C-R sample using robust location and scale estimates. The median was used as the location estimate, and the median absolute deviation about the median (MADM) was used as the scale estimate. Both estimates are know to be highly robust (Hampel,*et al*, 1986)[1] The second row of Figure 4 shows standardized data after taking the adjacent Control-Relevant pair and standardizing them as a single observation. By rescaling this way, we can make the relative differences (between the max of the control and max of the relevant, for example) easier to compare.

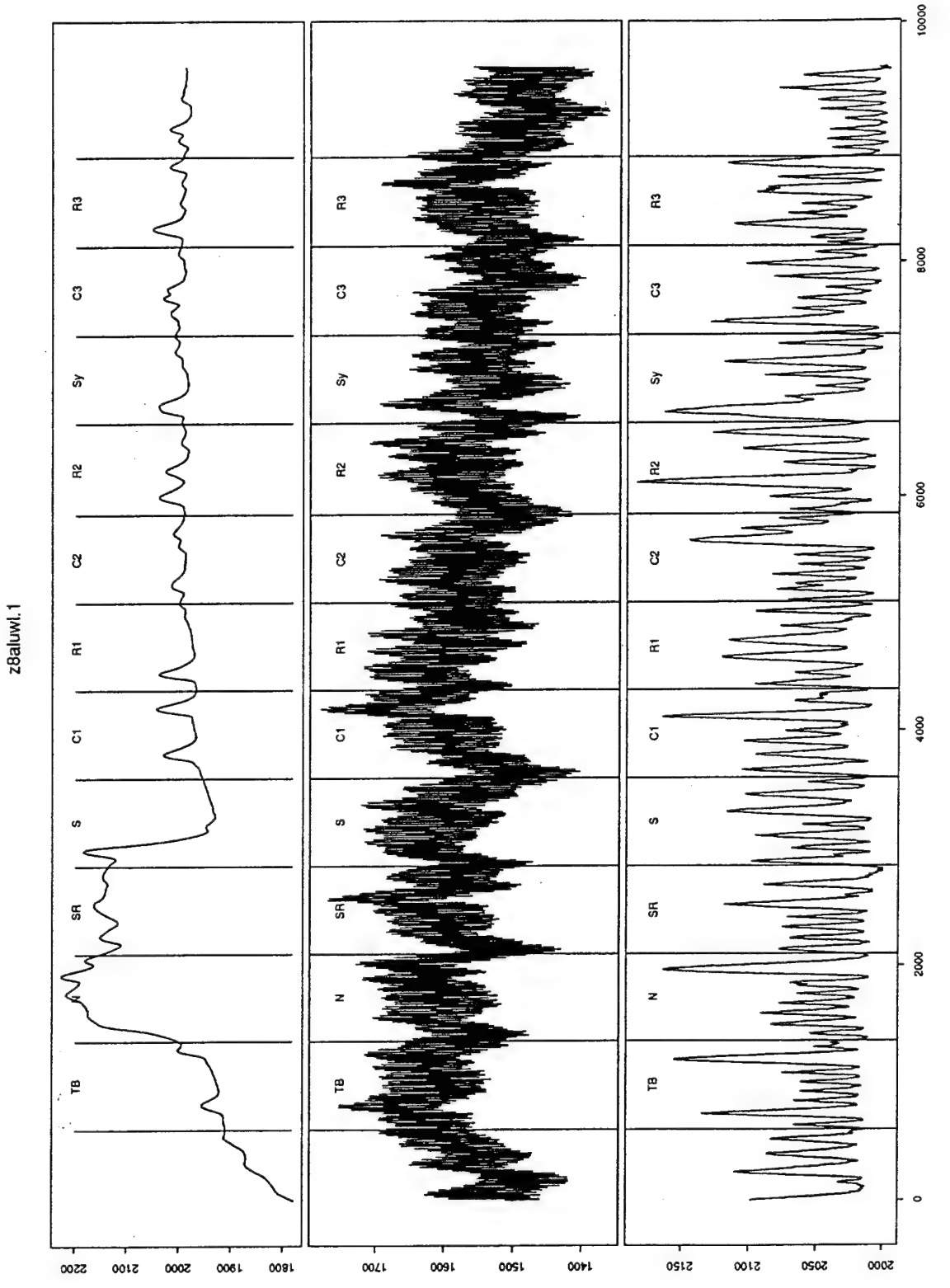

Figure 3: **Chart Plot Example**

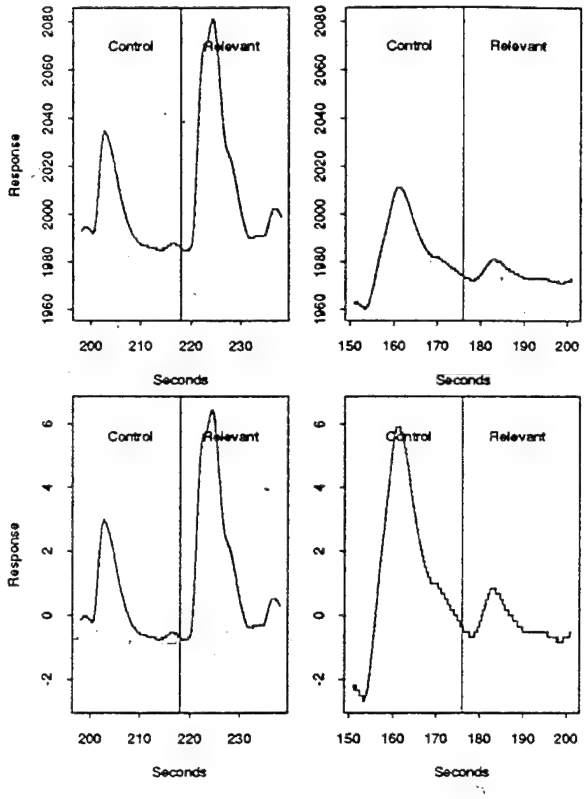

Figure 4: **Comparison of Standardization Methods**

Features

The features we use are grouped into two sets according to their origins. The first group consists of features that came from APL's work, plus a few more. These are quantities like medians and range estimates of the standardized series or of a detrended or differenced series. The second set of features is based on a spectral representation of the time series, which will be discussed in Section 4. These are the new features we are investigating.

Table 1 summarizes the features calculated on the data after it was standardized. Most of the APL and APL based features look at the location or variability of the time series or the first difference of the time series. We use the first difference to approximate the rate of change in the time series. Features 1–4, 15 and 16 explore the variability in the signal, the idea being that some subjects will have greater variability if they are being deceptive. We calculate features 5,6,11–14 based on the first difference of the series. Features 7 and 8 examine the smoothed series and take its first difference which reflects the overall trend in

the series and how it changes over time. Extreme values for features 9 and 10 can tell us that the smoothed series is very variable and hence the trend is a strong component in the series. Each of the first 16 features was calculated for all four channels.

The rest of the features 17-31, are based on spectral analysis. Features 17,18 and 22 were calculated using a single channel of information at a time, but because of their multivariate nature, features 19–21, and 23–31 require two channels. Because of the strong harmonic structure of cardio and the respiratory channels, only cardio and upper respiratory together and upper and lower respiratory together were used to calculate each of these bivariate spectral features.

After calculating the features for each C and R question separately, the control feature is subtracted from its appropriate relevant response feature thereby obtaining a "difference" feature for each C-R pair, except in the case of feature 22 which is discussed below. Since charts generally contain 3 C-R pairs, three difference features are typically computed for each chart. If only two C-R pairs are present, then we create a third difference feature by averaging the first two pairs. This was done to insure an equal number of responses from each chart which facilitates the CT method.

The Integrated Spectral Distance, feature 22, is calculated in a different way that the others by taking the standardized difference between the cumulative (summed) values of the unsmoothed periodogram for a control and the subsequent relevant question. Figure 5 shows the difference in a control-relevant pairing. Unlike the other features, this feature measures the distance between a control and a relevant pair directly in its calculation.

4 FREQUENCY DOMAIN

Univariate Time Series

The principle behind our use of frequency domain features is that a time series X_t can be approximately represented by a Fourier series with random coefficients at certain frequencies. This series can be represented in terms of sines and cosines or in terms of complex

Table 1: **Features Investigated**

Feature Number	Feature Description
1.	Maximum Value Achieved
2.	Minimum Value Achieved
3.	75%ile-25%ile from 2-14 seconds
4.	90%ile-10%ile from 2-14 seconds
5.	80%ile-20%ile of first difference from 0-8 seconds
6.	80%ile-20%ile of first difference from 0-4 seconds
7.	Median from 0-8 seconds of first difference of smoothed series
8.	Median from 8-end seconds of first difference of smoothed series
9.	25%ile of first difference of smoothed series
10.	75%ile of first difference of smoothed series
11.	Median of first difference from 0-2 seconds
12.	Median of first difference from 0-5 seconds
13.	Median of first difference from 0-8 seconds
14.	Mean of first difference
15.	Maximum-Minimum
16.	Variance
17.	Fundamental Frequency of detrended series (smooth trend removed)
18.	Maximum Spectral value of detrended series
19.	Phase at Fundamental Frequency
20.	Spectral value at Fundamental Frequency
21.	Coherency at Fundamental Frequency
22.	Integrated Spectrum Distance
23.	Spectral value at fundamental frequency of Channel 1
24.	Spectral value at fundamental frequency of Channel 2
25.	Spectral value at (fundamental frequency of Channel 1) * 2
26.	Spectral value at (fundamental frequency of Channel 1) * 3
27.	Spectral value at (fundamental frequency of Channel 1) * 4
28.	Coherency at fundamental frequency of Channel 1
29.	Coherency at fundamental frequency of Channel 2
30.	Coherency at (fundamental frequency of Channel 1) * 2
31.	Coherency at (fundamental frequency of Channel 1) * 3

Times are measured from the beginning of the question.

A rigorous version of the approximate equation (1) is known as the *Spectral Representation Theorem.* It is necessary to introduce the concept of covariance before we discuss this theorem.

The *covariance* of two random variables is a measure of their association or relatedness given by

$$Cov(X,Y) = C_{x,y} = E[(X - E(X))(Y - E(Y))].$$

In order to compare covariances, it is necessary to transform them to a standardized scale. We do this by dividing by the product of the standard deviations of X_t and Y_t and call this the correlation.

$$corr(X,Y) = \rho_{x,y} = \frac{C_{x,y}}{\sigma_x \sigma_y}.$$

The random variables X and Y are are said to be *uncorrelated* if

$$E[(X - E(X))(Y - E(Y))] = 0$$

For time series, the covariance is generalized to the *autocovariance* function

$$C_{t,t+l} = Cov(X_t, X_{t+l})$$

The time series X_t is said to be *covariance stationary* if the expected value of X_t has a constant value μ for all t, and $C_{t,t+l}$ depends only on l. We then write $C_l = Cov(X_t, X_{t+l})$. Then, the *autocorrelation* function:

$$\rho(l) = \frac{C_l}{C_0}$$

is the correlation coefficient for X_t and X_{t+l}.

Theorem: (Spectral Representation)

Let X_t be a covariance stationary time series. Then there exists an complex stochastic process $Z(\lambda)$, $\lambda \in [0,1]$ having stationary orthogonal increments such that X_t can be written as

$$X_t = \int_{-\frac{1}{2}}^{\frac{1}{2}} e^{i2\pi\lambda t} dZ(\lambda)$$

and for $-\frac{1}{2} \leq \lambda_1 \leq \lambda_2 \leq \frac{1}{2}$

$$Var(Z(\lambda_2) - Z(\lambda_1)) = F(\lambda_2) - F(\lambda_1)$$

where F is a nondecreasing function with $F(1) = Var(X_t) = \sigma_x^2$.

The process $Z(\lambda)$ has orthogonal increments if the correlation is zero for non-overlapping intervals of $Z(\lambda)$, $\lambda \in [-\frac{1}{2}, \frac{1}{2}]$. So

$$E[Z(\lambda_4) - Z(\lambda_3))(Z(\lambda_2) - Z(\lambda_1)] = 0 \qquad \text{if } \lambda_1 \leq \lambda_2 \leq \lambda_3 \leq \lambda_4$$

This function $F(\lambda), \lambda \in [0, 1]$, is called the *spectral distribution* function of X_t, and its derivative f is the spectral density function:

$$F(\lambda) = \int_0^{\lambda} f(x)dx$$

$$f(\lambda) = \frac{dF(\lambda)}{d\lambda}$$

Using the spectral representation it is straightforward to establish the following well known Fourier transform pair relationships between the autocovariance function C_l and its spectral density $f(\lambda)$

$$C_l = \int_0^1 f(\lambda)e^{i2\pi l\lambda}d\lambda, \qquad l = 0, \pm 1, \pm 2 \pm 3 \ldots$$

$$f(\lambda) = \sum_{l=-\infty}^{\infty} C_l e^{-2\pi i l\lambda}, \qquad \lambda \in [0, 1].$$

Estimation of $f(\lambda)$

A very common way to investigate a spectral density function $f(\lambda)$ is to compute a smoothed periodogram estimate following the steps below:

1. Detrend the data.
2. Apply a "data taper" to reduce leakage.
3. Compute the Fourier coefficients $X(\lambda_k)$ using the FFT (Fast Fourier Transform).
4. Compute periodogram $P(\lambda_k) = |X(\lambda_k)|^2$.
5. Smooth the periodogram to reduce variance

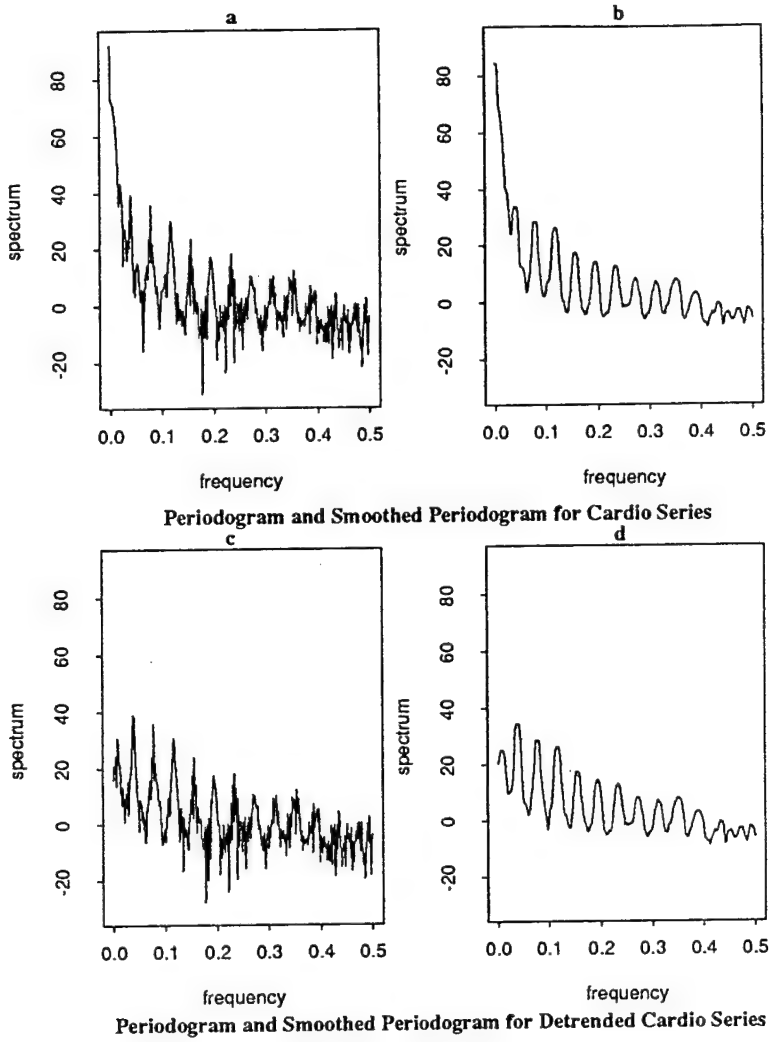

Figure 6: **Smoothing and Detrending**

The initial detrending step is needed to remove low frequency energy that may reduce our ability to see other features, e.g., lower energy cycles in the series. Typically we will use a smoother with a variable span to remove any long term trends that are in the data.

Figures 6a-b show the raw periodogram $P(\lambda_k)$ and a smoothed periodogram $\hat{S}(\lambda_k)$ for the cardio channel time series, with the detrending step 1. Notice how the smoothing used in computing $\hat{S}(\lambda_k)$ reduces the noise evident in $P(\lambda_k)$. Figures 6c-d show the same quantities as in Figures 6a-b, except that now the detrending step is included. Note the removal of the low frequency energy which dominates Figures 6a-b.

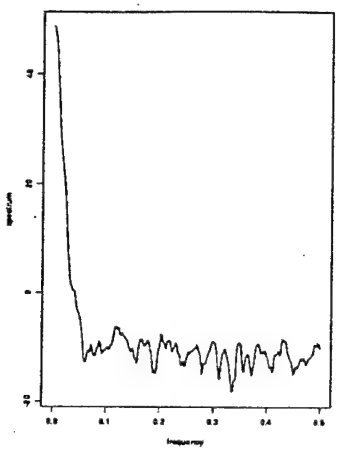

Figure 7: **GSR Smoothed Periodogram**

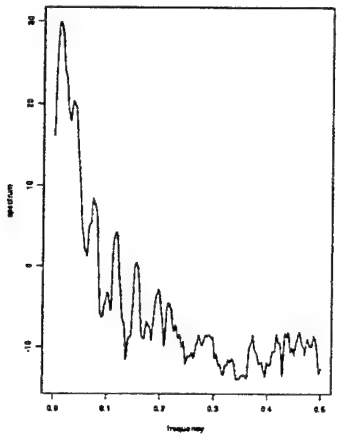

Figure 8: **Lower Respiratory Smoothed Periodogram**

The second data tapering step helps us to smooth out leakage from neighboring frequencies. A 10% cosine taper is applied to both ends of the series to reduce this effect. See for example see Bloomfield, 1976[2], for information on tapering.

The Fast Fourier Transform gives a decomposition of our detrended and tapered time series into orthogonal sines and cosines. The periodogram computed in step 4 is an unbiased estimate of the true but unknown spectral density $f(\lambda)$. However, the variance of $P(\lambda_k)$ does not go to zero as the sample size/series length increases and so smoothing is needed to reduce variability.

Note that the spectrum estimate for the Cardio series has a distinctive structure that corresponds to the strongly periodic character of the series. The main feature in this figure is that the peaks appear to be evenly spaced. The main peak occurs at the *fundamental frequency*, and the other peaks are at or near integer multiples of the fundamental. These "echoes" of the fundamental are called the *harmonics* of the series. They help describe the deviations of the series away from the sinusoidal pattern determined by the fundamental.

The GSR series spectrum is shown in Figure 7. Notice that most of the variability is coming from the very low frequencies, and there is no evidence of periodic behavior. This means that the GSR series is mostly trend like in nature.

Figure 8 is a typical spectrum from the lower respiratory series over the same question as the cardio series in Figure 6d. This series has been detrended, but a very strong low frequency oscillation remains. This same frequency appears on the cardio above which suggests that this signal is induced by the respiratory process on the cardio-vascular process. We also see some evidence of the higher frequency harmonics in the respiratory process.

We are interested in measuring how different a control question response is from a relevant question response, in terms of spectral densities. One common approach is to construct a distance between spectral distribution functions (integrated spectral densities) as follows. The Integrated Spectral Densities (ISD) plots for the cardio series are shown in in Figure 5 a control (F_c) and a relevant (F_r). These are the cumulative sums of the periodograms, $F_c(\lambda_k) = \sum_{j=i}^{k} P_c(\lambda_j)$ and $F_r(\lambda_k) = \sum_{l=i}^{k} P_r(\lambda_l)$. We can apply different distance measures to determine how these two ISD's differ, that is, how far apart the control and relevant are from each other. A common method is to use the maximum vertical difference between the two ISD's:

$$d(F_c, F_r) = \max_{\lambda} |F_c(\lambda) - F_r(\lambda)|$$

S-Plus makes calculation of these terms fairly simple. The `spectrum()` function calculates the smoothed periodogram including the data taper. The `supsmu()` function was used as a the detrender.

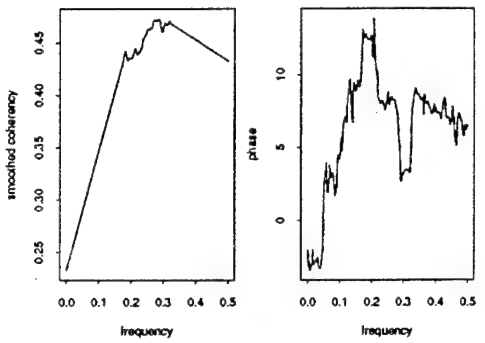

Figure 9: **Phase and Coherency**

In addition to looking for phase relationships between the components series of a multivariate time series, one maybe quite interested in possible correlations between the series at different frequencies λ. such correlations are captured by a quantity know as the *coherency* function $W_{j,k}(\lambda)$ for the two series $X_{j,t}$ and $X_{k,t}$. The squared value of the coherency is easily represented in terms of the spectral densities $f_j(\lambda) = f_{j,j}(\lambda)$, $f_k(\lambda) = f_{k,k}(\lambda)$, and the cross spectral density $f_{j,k}(\lambda)$ for time series $X_{j,t}$ and $X_{k,t}$, namely:

$$W^2_{j,k}(\lambda) = \frac{f_{j,k}(\lambda)^2}{f_{j,j}(\lambda) f_{k,k}(\lambda)}$$

The coherency function has a very simple interpretation as the correlation between two series at frequency λ. The way to think of this is in terms of the two complex Fourier series representations:

$$X_{j,t} \approx \sum_{n=-N}^{N} B_{j,n} e^{-i2\pi\lambda_n t}$$

$$X_{k,t} \approx \sum_{n=-N}^{N} B_{k,n} e^{-i2\pi\lambda_n t}$$

For the two series $X_{j,t}$ and $X_{k,t}$, $W_{j,k}(\lambda_n)$ is the correlation coefficient between $B_{j,n}$ and $B_{k,n}$. An example of the estimated coherency and phase between the cardio and lower respiratory

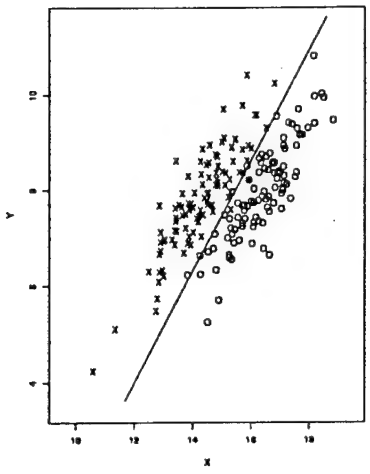

Figure 10: **Example of Linearly Separable Data**

channels is given in Figure 9. The coherency has been smoothed to reduce variability. Phase and coherency are given calculated using the Splus function `spectrum()` which adjusts the phase to be more continuous by removing all jumps of of greater than pi.

5 CLASSIFICATION TREES

Fisher's linear discriminant analysis is one of the most common classically based tools for classification. The Fisher discriminant works quite well when two categories are separated linearly. An example of this is seen in Figure 10. If however, the data requires a non-linearly function for good separation, then Fisher's discriminant will not be very successful in partitioning the data. This can be seen in Figure 11. If we look at Figure 12, we see how a nonlinear rule can considerably dominate the linear rule. This motivates us to use the modern *classification tree* (CT) to determine deception or non-deception.

The basic idea behind the classification tree method is that the data is partitioned in such a way so that within each partition the data is made as homogeneous as possible with respect to the classifying variable. This method of recursive partitioning lends itself naturally to a tree representation as shown in Figure 13. First a tree is "grown" and then it is "pruned" to a smaller size. Each of the partitions is called a "node" of the tree.

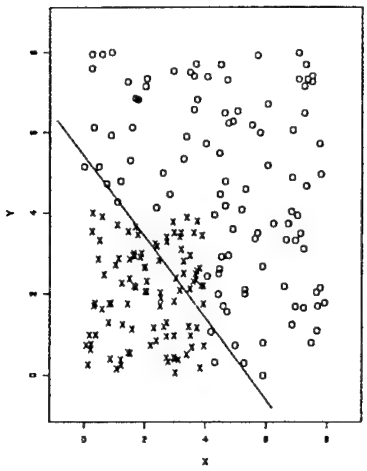

Figure 11: **Example of Non-linearly Separable Data**

In Figure 12, the CT method first chooses a partition along the X axis and then divides the left half ($X < 3.98$) into two sections by ($Y < 4.05$). At this point, each of the regions is homogeneous and hence no more splits are necessary. If more than two predictors are available, then the tree representation is used since it is not possible to present the splits in higher dimensions.

The tree representation gives us more detailed information at each of the nodes. Looking at Figure 13, the value inside the node is the value held by a majority of the the observations at the node. The fraction below counts the number misclassified divided by the total number at the node, so at the top (or "root") node, we have 200 observations 100 of which are X's (misclassified) and the other 100 are O's. If X is less than 3.98 then we choose the left branch. Since 29 of the 129 observations are O's, the algorithm searches to find a split. We discover that when Y is less than 4.05 we can get a pure node of all 100 X's and another node of 29's Y's. If we had $n > 3$ variables, then we would be forced to use the tree representation, since we could not draw an n dimensional space.

In order to guide the partitioning toward the homogeneous state, we need to design a function that measures the "impurity" at a node. One natural way to construct an impurity function is given below. A standard model for binary response data is that of the binomial

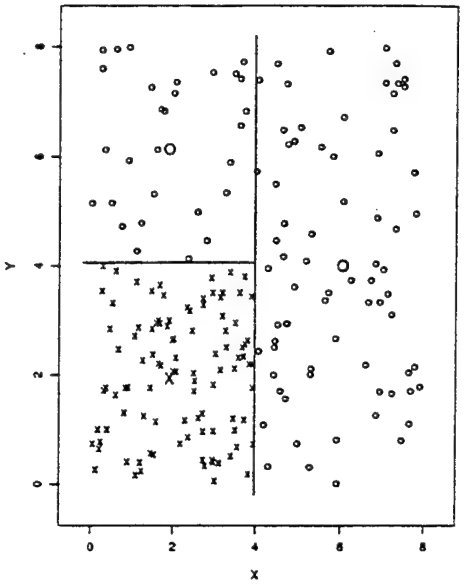

Figure 12: **Classification Tree on Non-linearly Separable Data**

distribution which has the following for -2 log-likelihood for each observation:

$$-2\left[ylog(p) + (n-y)log(1-p)\right]$$

where p denotes the probability that a subject is deceptive, y is the number of deceptives, and n is the total number of observations at the node. Since the p's are unknown, we estimate them by $\frac{y}{n}$ at each of the nodes. This -2 log-likelihood term is known as the deviance and is a common measure of goodness of fit for a model. In this case, the deviance is minimized when p is 0 or 1 and is maximized at 0.5. This function is also concave down with a unique global maximum. The CT algorithm searches for splits along a single variable that will minimize the impurity as measured by the deviance. The S-Plus tree model is the specific implementation of the CT algorithm that we are using (Chambers and Hastie, 1992)[4].

Some of the more technical rules that govern the building of CT models are discussed in the appendix.

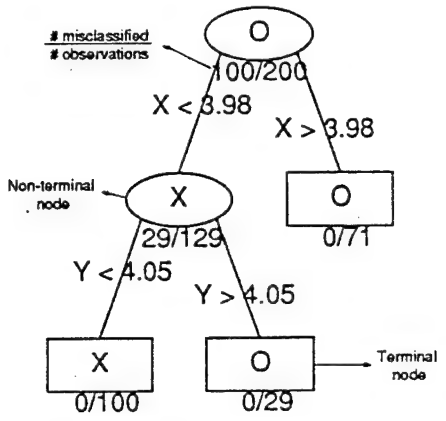

Figure 13: **Tree for Figure 12 Splits**

6 EXPERIMENTAL METHODS

Many different classification experiments were performed in an effort to find the best way to use the classification tree approach and to deal with the very large dimensional feature vectors one might use. Three main experiments based on different choices of feature vectors will be reported on below. The first experiment uses only a few GSR features, and the second adds some frequency domain features. The third experiment uses all the features reported.

Our current tree modeling method employs two classification trees. The first tree is built by classifying each of a set with 567 charts from all subjects under consideration according to deception(D) or no deception (ND). A cross-validation step is used to select the proper sized CT model according to generally accepted statistical practices. This final model gives fitted scores between 0 and 1 for each of a subject's charts. The fitted score for each subject at a single node is the mean value of the subjects' response variable at that node. If there are 20 charts at a node and 5 are known D (coded as 0) and the other 15 are ND (coded as 1), then the fitted score at the node is $15/20 = 0.75$.

To combine the different charts, the mean of these fitted scores is taken. Because these charts undergo an averaging step, it is not always the case that a mean score above 0.5 will indicate ND or a mean score below 0.5 will indicate D. This leads to building a second tree grown from the set of subject mean scores. After undergoing a cross-validation step, we see

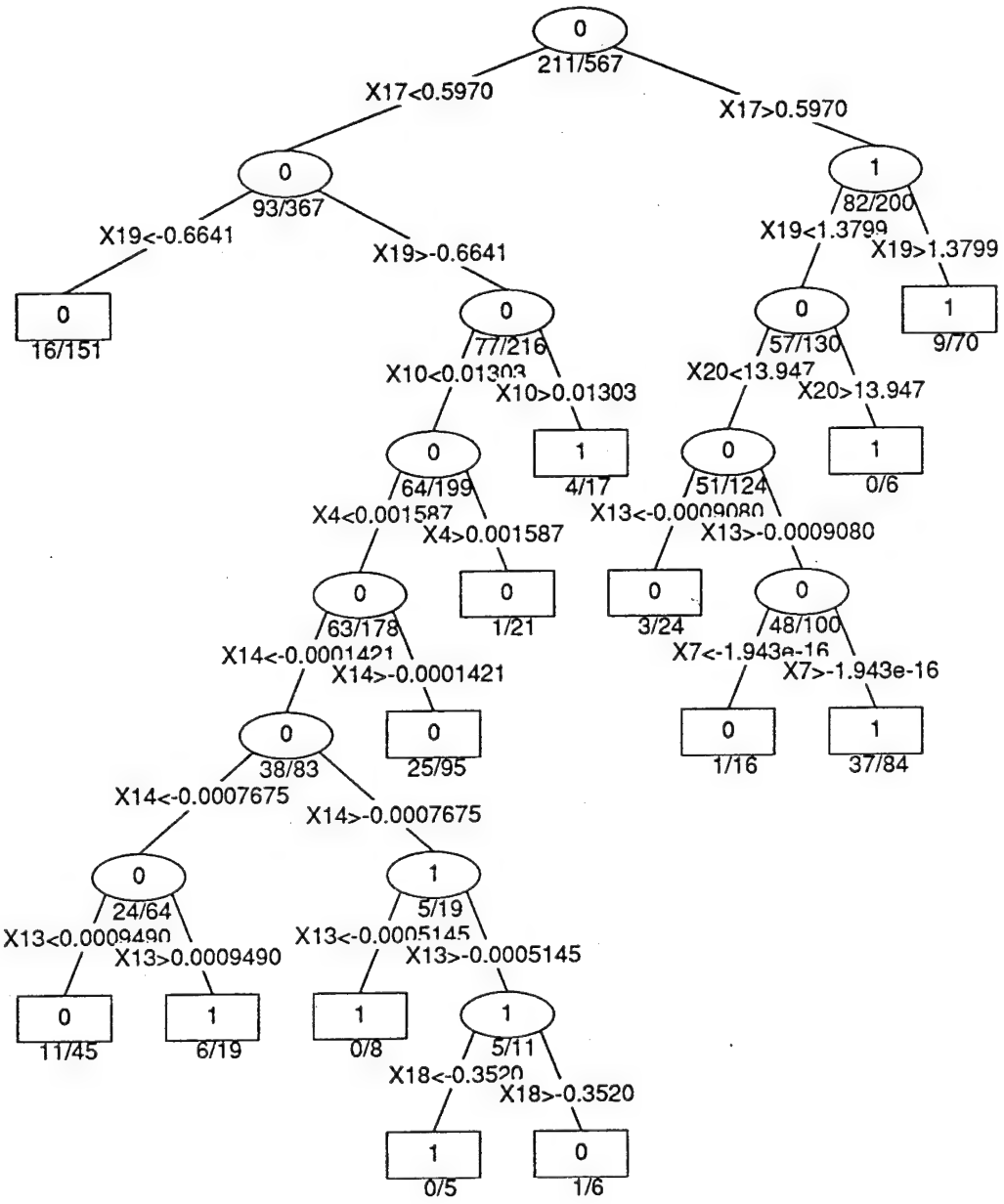

Figure 14: **Chart Tree for Experiment 1, Full Data**

Table 4: **Variables in Final Tree Model-Experiment 1**

Full Data	A Data	B Data
Range of GSR(2)	Range of GSR(2)	Range of GSR(1)
Variance of GSR(1)	Variance of GSR(1)	Median 1^{st} diff 0-8 sec GSR(1)
Median 1^{st} diff 0-8 sec GSR(1)	Variance of GSR(3)	Mean of 1^{st} diff GSR(2)
Median 1^{st} diff 0-2 sec GSR(1)	Range of GSR(1)	Variance of GSR(1)
Mean of 1^{st} diff GSR(2)		
Mean of 1^{st} diff GSR(1)		
Range of GSR(3)		
Var of GSR(2)		
Median of 1^{st} diff 0-5 sec GSR(1)		

Numbers in parenthesis indicate which of the three differences was chosen.

Table 5: **Misclassification Error Rates-Experiment 2**

	Train Full	Train A	Train B
Test Full	0.151	0.181	0.226
Test A	0.115	0.123	0.344
Test B	0.208	0.273	0.039

Variance of GSR, then on feature 13 calculated for GSR.

6.2 Experiment 2

Experiment 2 was motivated by our desire to show the power of spectral features. For this, we calculated features 23-31 using cardio and upper respiratory, and then calculated them again using upper and lower respiratory. We also included the standard GSR features 1-6 and the features used in Experiment 1. The first splits in the final tree included feature 3 for GSR and features 25 and 27 calculated at the fundamental of UR. The results are summarized in Tables 5 and 6.

Notice that the B data does very well on predicting itself, but is very poor at predicting the A data.

Table 6: **Variables in Final Tree Models -Experiment 2**

Full Data	A Data	B Data
0.75-0.25 from 2-14 sec GSR(1)	0.75-0.25 from 2-14 sec GSR(1)	0.9-0.1 of 2-14 sec GSR(1)
0.75-0.25 from 2-14 sec GSR(2)	Variance of GSR(2)	0.8-0.2 1^{st} diff 0-8 sec GSR(1)
Spectrum of UR at $ffur$ *4	0.9-0.1 from 2-14 sec GSR(2)	25%ile 1^{st} diff smo LR(1)
25%ile 1^{st} diff smo GSR(3)	0.8-0.2 1^{st} diff 0-4 GSR(3)	Median 1^{st} diff 0-8 sec GSR(1)
25%ile 1^{st} diff of smo UR(3)		Fund freq of LR(3)
0.9-0.1 of 2-14 sec GSR(2)		Mean of 1^{st} diff GSR(2)

Smo indicates smoothed, 0.9-0.1, 0.75-0.25 and 0.8-0.2 all indicate that the difference of these quantiles were reported. Fund freq is the fundamental frequency of the series measured by the frequency at the maximum of the periodogram.

Table 7: **Misclassification Error Rates-Experiment 3**

	Train Full	Train A	Train B
Test Full	0.126	0.121	0.201
Test A	0.098	0.057	0.311
Test B	0.169	0.221	0.026

6.3 Experiment 3

Experiment 3 combined all the features that we had calculated to date. This included features 1–10 calculated on all four channels; features 11-16 calculated only for GSR; features 17-21 calculated on Cardio, UR and LR; feature 22 is calculated only for GSR; and finally features 23-31 are done the same as in Experiment 2, using the two bivariate series (Cardio,UR) and (UR,LR). When the whole dataset is put through the chart tree, the top splits are on feature 3 for GSR, feature 27 for UR at UR's fundamental frequency and feature 3 for UR. The results are given in Tables 7 and 8.

Again, the B data does very well on predicting itself but does not do well at predicting A. Overall, the 12.6% error rate is fairly good considering the differences between the two datasets.

Table 8: **Variables used in Final Tree Models-Experiment 3**

Full Data	A Data	B Data
0.75-0.25 from 2-14 sec GSR(1)	0.75-0.25 from 2-14 sec GSR(1)	0.9-0.1 of 2-14 sec GSR(1)
0.75-0.25 from 2-14 sec GSR(2)	Variance of GSR(2)	0.75-0.25 of 2-14 sec of UR(1)
Spectrum of UR at ff_{UR} *4	75%ile of 1^{st} diff of smo UR(3)	0.8-0.2 1^{st} diff 0-8 sec GSR(1)
0.9-0.1 from 2-14 sec GSR(2)	Med 1^{st} dif 8-end of smo GSR(2)	Fund freq of smo GSR(1)
0.75-0.25 from 2-14 sec of UR(3)	Phase at fund(UR) 0-8 sec UR(2)	Fund freq of Cardio(2)
0.8-0.2 of 1^{st} diff 0-8 sec Cardio(3)	Minimum Cardio(3)	0.8-0.2 of 1^{st} diff 0-4 sec Cardio(2)
Fund freq UR(1)	0.9-0.1 from 2-14 sec UR(1)	75%ile of 1^{st} diff of smo UR(2)
		25%ile of 1^{st} diff of smo LR(3)

7 Conclusions

The first year's work on this project has lead us to look at several issues. First, we think that GSR is a very important channel and investigating its features in the frequency domain should yield interesting results. On the data organization side, a site visit showed us that there is an enormous amount of editing and cleaning that should be done before trying more experiments. In many cases movement has distorted the information that is available so that the algorithms may have been trained improperly.

The site visit also validated our trepidation in trusting the results of using the A and B datasets as test samples. Their origins are quite different, and hence results may not be generalized between the sets.

Although the cardio channel was not in the first three splits in any of the experiments, it was used in all of the final models when the complete feature vector was used (Experiment 3). We will try different smoothing techniques on the cardio channel to improve the effectiveness of frequency domain analysis.

We feel quite comfortable with many of our techniques using the classification tree, however future work will probably entail comparing different methods which combine the multiple charts from a single subject. This has a cost of increasing the dimensionality of the data, but we expect work in this area to be effective at error rate reduction.

Toward the end of the first year, we began to explore two new methods that might

explain variability in the data. The first method was the ISD–Integrated Spectral Distance. Because GSR is not harmonic in nature, it is very difficult to use features like the fundamental frequency to describe it. Instead, we can work with the complete spectral representation and measure the spectral distance in GSR between a control and a relevant question.

Our second new technique is to use a variance component model to perform harmonic regression. Harmonic regression works by estimating a set of two or three sine and cosine terms with frequencies determined by the fundamental frequency in the periodogram. In addition to these fixed terms, a set of random effects will be estimated for each cycle of the series which will represent random adjustment terms to the overall fixed terms. By examining the series of random effects, we can determine if an event has occurred in one of the cycles. The differences in the sequence of estimated random effects can then be used as new features to see if any changes took place over the course of a question. The S-Plus function `varcomp()` will be used to estimate the random and fixed coefficients.

8 Appendix on CT Models

The second part of the cost function for a CT is the penalty term for increasing complexity in the model which we will call λ.

When a node has nodes below it, then it is called the parent node and the sub-nodes are the children nodes. For each parent node, we calculate the cost function as

$$C(node_i) = C(child1) + C(child2) + \lambda.$$

If $node_i$ is a terminal node and has no children, then the cost for the node is just the deviance.

To find the total cost of the tree, we sum of the costs of the terminal node and add λ times the number of splits in the tree.

Notice that if λ is zero, then we could grow as many "branches" (sub-nodes) as we liked without any penalty. Also, if λ is very large, then we will not have an incentive to branch and so our model will have just one "root" node. In fact, there exists a sequence tree sizes which minimize costs that correspond to a sequence of λ's. Unfortunately, we cannot estimate the best tree size directly since the λ's that we get from the whole dataset are for a fixed cost.

2 Appendix II

The year two report prepared by Chris Pounds and Doug Martin.

Polygraph Reliability: Year Two Report: Part 1

Chris Pounds and Doug Martin

January 27, 1994

1 Overview

This document summarizes the work on the NSA Polygraph project from October 1992 through June 1993. The reader should be familiar with the first year report of this project which gave a basic outline of the polygraph problem and listed some of the feature vectors that have been calculated. Unlike the first year report, this report includes many details that are not critical to the extraction of good features. Much of the work in the second year has been with organizing the manipulation and translation of the data, so some sections may be skimmed without great loss for a reader who is concerned about specific features.

First, we review what work has been done, and then examine the work chronologically beginning with October of 1992. October's and November's work dealt with verifying the integrity and quality of the charts in the S-PLUS database. In December we used an Object Oriented paradugm to streamline the functions used to build features in the first year. January and February were data management months—the previous year's work was condensed and organized on the computing systems. In March, we received MGQT files which were converted from Axciton format into S-PLUS format. The March through June work involved integrating new charts into the database. Finally, we discuss specific improvements to our methods for the Zone charts.

Because different computer environments are mentioned, we need a convention to refer to

computer related objects. Computers in the Department of Statistics are referred to in the courier font. These machines are paros, grover, and sparta. Files and directories on these machines will also be referred to in courier font. S-PLUS is the principal environment for analysis of this work, and functions and objects in S-PLUS are referred to as function() and object with the sans serif font. UNIX and DOS commands are in *italics*. Any UNIX executable script file will also be in *italics*.

2 History

In March 1992, Chris Pounds traveled to Johns Hopkins University's Applied Physics Lab in Columbia, Maryland, to learn about the polygraph project. He returned with floppy disks containing the A and B "January Parameters" calculated by APL. These parameters were a collection of 95 features calculated for each question on each chart on about 200 subjects. The data contained an uneven number of fields which meant that a C program needed to be written to process the files. During this period we learned how to handle large files and how to use the tree() and associated modeling functions in order to use a tree classifier for discriminating between the deceptive and non-deceptive subjects. Work from this early period is contained in directory `sparta:/home/clbs/ra/abdata` and its subdirectories.

In June of 1992, Chris returned to APL and obtained copies of the charts in the Axciton format. These "raw" datafiles were put on floppy disks and compressed using pkzip, a compression format that is accessible in both UNIX and DOS. There was also work during this period on a program for processing charts into ASCII format given the Axciton data. Phil Neal was a big help in finding a C function called *swab()* for performing byte swapping. The program also translated the DOS file format into the UNIX file format particular to the SUN workstations that the Department of Statistics uses. The resulting program is called `dec.c` and can be found in `paros:/users/clbs/ttemp/dmgt`.

Work during July and August focused on getting the tree models running. Originally all standardization of a series with respect to the location and scale parameters was performed on a per question basis so that after standardization the time series for all question had the same location and scale estimates. Simple plots in the first year report show that the

single question standardization is not correct because it washes out differences between the C-R (control and relevent) question pairs. Instead, standardization (using the median and median absolute deviation) was perfromed on each C-R question pair. This seemed to be fairly effective at maintaining the differences between pairs of questions, and also meant that all the previous calculations were obsolete. Between September 1 and September 17 (the day of NSA's visit), we worked on getting as many parameters and models through the test procedure as we possibly could. The datasets were collections of six or eight features that had been calculated using a single feature extraction function. The Year One Report summarizes our findings.

3 October 1992 work

3.1 Summary

We identified charts with movements or other anomalies using two methods:

1. By displaying each chart using S-PLUS and examining it visually for jumps or severe outliers.

2. By reading each chart's question text and verifying that the question pattern matched the pattern that was automatically assigned.

3.2 Details

APL and NSA labeled the datasets according to their delivery dates to APL. The "A" dataset consisted of 129 subjects received at APL before May of 1992, and the "B" dataset consisted of 90 subjects received in May and June of 1992. Within these groupings, the following source groups were separated into different directories: FBI, Clayton County, Vermont State Police, SLED, DEA, Marion County, Birmingham and Anniston.

When subject files were received, they are associated with their original Axciton coding name which is given by a character string with two leading $$ symbols followed by a number, then a character string of length 5. A suffix of length 3 was also used to identify files contents.

The suffix has a leading 0, then a number to indicate the chartnumber, *i.e.,* 1, 2, 3, 4, or 5), then a 1 to indicate the Event Marker file, a 2 to indicate the Chart Datafile, or a 3 to indicate the Question Text file. Examples of this are $$4r%kto.031 (subject 4r%kto's Event Markers for his 3rd chart) $$4ukdf#.012 (subject 4ukdf#'s Chart Data for his 1st chart). We had collected information on the A and B datasets earlier in 1992 which was used to match the file labels. Some of the charts were improperly labeled, presumably from having the exam restarted in the middle of a session or from being mismatched with their responses, so the A dataset was trimmed to 122 subjects and the B dataset had only 77 usable subjects. Of the 122 A subjects, 76 were deceptive and 46 were non-deceptive. For the 77 B subjects, 49 were deceptive and 28 were non-deceptive. Deceptive subjects accounted for about 63% of the subjects in each group.

	Subset A	Subset B
Deceptive	49	76
Non-deceptive	28	46

	Number of Charts				Total #
	1	2	3	4	of Charts
A subjects	8	15	85	14	349
B subjects	3	10	59	5	220

When an examiner administers the Zone format exam, there is a fixed ordering of questions. On some occasions, the examiner will deviate from this fixed format, usually, by making a notation in the Question Text file. This leads to a strange pattern in the event markers. Instead of "0 1 2 0" marking the beginning, ending and response of a question, the 2 or 1 will be missing, repeated, or transposed in the series. We applied the standard ordering to all of the charts and initially only looked at the event markers to make sure they matched our automated marking. When the event marker list for a chart did not agree with the automated method, we examined the original question texts to see what the difficulty was. For further verification, we examined each question text to see if it matched the automated question ordering. This was done for nearly 600 charts, a simple but lengthy task.

Another part of our examination of data quality was the visual examination of each chart. We discovered that some of the charts contained anomalous artifacts such as large jumps

in all four channels or a substantial fall-off of the GSR channel. The examiners informed us that these are caused by subject movement during the exam and therefore these exams might be difficult to score algorithmically. We decided to keep these charts in the database for two reasons. First, we could not be sure that they were actual movements. There was a chance that some of them could have been genuine features that we could use in our analysis. Second, by leaving them in and identifying them, we could determine the impact of charts movements on the algorithm by performing a sensitivity analysis. This required us to repeat our model building process without the questionable charts and then investigate the differences between the two modeling processes.

The following is a listing of the 75 charts that are suspected of containing gross artifacts.

z4r5eo.1	z4rkto.1	z4u5kji.2	z4umj1i.3	z4xkuqu.1
z4xkuqu.2	z4xyakf.2	z6efai.1	z6trwi.1	z79nbu0.1
z7b0olx.1	z7cmxc.1	z7cmxc.2	z4pbmai.2	z4z0fa3.1
z4z15v.3	z6e5kb9.1	z6j06ta.1	z6ldyhl.1	z6ldyhl.2
z6ldyhl.3	z5sdwx.1	z5sdwx.2	z6ogei.1	z6ogei.2
z6onkb.2	z5y4ui3.1	z5y4ui3.3	z6xju.3	z7ow8bi.2
z7ow8bi.3	z7plt4i.3	z7zegu0.1	z7ysr3.1	z7ysr3.2
z7ysr3.3	z7x44m.1	z831sp3.3	z85o2i3.4	z8ayvz.1
z8fdy3f.2	z8k78pc.2	z81yox.1	z86slr.1	z86slr.4
z86slr.5	z87zh4s.2	z888rmb.4	z8al2gx.1	z8al2gx.3
z8al2gx.4	z8aw85r.1	z8aw85r.2	z8jxfxa.1	z8jxfxa.2
z8jxfxa.3	z8layx.1	z8layx.4	z8lanz9.1	z8mfxdl.1
z84jaer.1	z84jaer.2	z84jaer.3	z87k6u.1	z8cuijr.2
z8qy8dl.1	z8lfqc.1	z8lfqc.2	z8lfqc.3	z8nm9tr.1
z8nm9tr.2	z8oekyc.2	z8s1f0.1	z8s1f0.2	z8sg7nc.2

Of the 75 badcharts identified, 44 are deceptive charts (59%) and 31 are non-deceptive charts. It seems reasonable that deceptive people would move more often than non-deceptives simply because they will try to "beat the test."

4 December 1992 Work

4.1 Summary

Functions were rewritten to take advantage of changes in S-PLUS. These changes make it easier to add new features to the feature vector and also take up less space. Functions for

building the features and manipulating the data in the tree method stage were updated.

4.2 Details

S-PLUS is not designed to manipulate extremely large data objects (> 4 Mb) like a database language can, but S-PLUS is quite efficient in working with small to modest size files (< 0.5 Mb). With this in mind, we have arranged the database so that each chart is stored as an individual file. Functions we wrote can access the charts either one chart at a time or one subject at a time.

Originally, we built functions which could be written quickly and with only a rough skeleton. At the end of the first year's work, we examined the functions and exploited the object oriented paradigm to set up an environment which allowed for efficient and modular code designs.

In the first version of what we call "feature extraction", the function `npc()` (new prepare chart) was called with the following arguments:

```
npc(subjectname, columns, function, nargs)
```

subjectname:	name of the subject being examined
columns:	which channels of the chart are to be extracted (GSR=1, Cardio=2, UR=3, LR=4)
function:	which feature extraction function set would be used
nargs:	number of features that function calculates

Within `npc()` the structure looked like this:

```
begin function
calculate the number of charts for the subject
create storage for each of the features that the feature extraction
        function produces
begin loop over each of the subject's charts:
        call feature extraction function for appropriate channel
        store the question labels and features
end loop
begin loop over charts
    begin loop over features
        calculate differences between control and relevants
        if only two CR pairs, calculate third as average of first two
    end loop over features
```

```
end loop over charts
return matrix of differences
end function
```

The feature extraction functions (FEF) had the structure:

```
begin function
find the question marker file
figure out which rows of the chart contain the control and relevant questions
        ( e.g., 500-1000,1200-1900, 2000-2600)
pair up the control and relevant questions
begin loop over each CR pair
        find subset of chart that contains CR pair
        standardize CR pair together
        calculate features for control
        calculate features for relevant
        store features
end loop
return features for each c and each r
end function
```

In our end of the year examination of these functions we saw two limitations that could be overcome by using object oriented programming. The first negative was that the code for determining the rows containing control and relevant questions was duplicated in each FEF. The use of a single function would eliminate this repetition. In fact the FEF's could be separated into two classes according to the number of channels they processed—many of the features are calculated on one channel at a time, but the multi-spectral features were done on two channels at a time. This lead to the construction two different "generic" object oriented extraction functions called caller.uni() and caller.biv(). These two functions solved the repetition problem. Specific calculation functions were then written to correspond to features that were calculated in previous work.

The second problem was that you needed to know the number of features that an FEF was going to calculate in order to call the function from the subject level npc () function. In order to solve this problem, a "nargs" attribute was added to each FEF. This attribute is used to tell functions using the FEF the number of features that the FEF creates. In addition to "nargs", each FEF was given a "class" of either "uni" (for univariate) or "biv" (for bivariate), allowing functions using the FEF the ability to determine which of the "caller" functions to use.

The class construct allows one to use a generic function call which will take the calculation function as an argument. The generic function caller() will evaluate the class attribute of the calculation function and will then use the class as an extension to call the appropriate caller function. For example, the class attribute of the calculation function called call.gs1() (which calculates 4 first difference features plus the range and the variance) is "uni," so caller(call.gs1) will evaluate as caller.uni(call.gs1). In addition, caller.uni() will automatically extract the "nargs" attribute of call.gs1() (which is "6") to allocate a matrix of the proper dimension to store the results.

The final product of this endeavor is that new features can be added by only having to write a calculation function with a couple of simple attributes. Similar work on plot functions is much more difficult because of the complication of setting global plotting parameters such as the range of the data.

Once the feature vector is created it must be bound together in an S-PLUS matrix-like data object called a "data frame." In the data frame each row contains the features (combined from several FEF's) for a single chart. Also included in the data frame is information regarding whether or not the subject was deceptive. The tree classifier can then be applied to each chart, and a prediction was obtained from each chart regarding whether or not the subject was deceptive. These predictions can then be combined in a separate tree procedure using the chart level predictors, and the final subject based (as opposed to chart based) predictions regarding deception can be obtained.

Prior to December, we had needed to create new functions and data objects for each new feature vector since we could not depend on the same charts being in every feature vector. In December, we discovered that the functions we had been creating were quite similar, so we went to work on making them more interchangable (generic).

The key step in making interchangable functions is to make all the arguments depend either on global parameters that do not change or on features that can be directly extracted from the working data frame. When revising old functions, we would change only a few of the statements in the function. While the old function would extract information from the data frame's global name, for example, the new functions used arguments to obtain this

information.

Functions are listed at the end of this report.

5 January and February 1993 work

5.1 Summary

We reorganized the file systems to allow for better access of files—old files that did not contain pertinent information were removed and important files were grouped and organized.

5.2 Details

Physical scientists are taught to keep a journal of what occurs in their study environment. As a statisticians, we can use the files that we have created as a record of our progress. Over the course of 18 months, the file structures can become very complex. When an analysis method is repeated or modified, the previous record may not be essential and is overwritten or simply deleted. This action may be transparent to the analyst. However, someone else who examines the files may not see these hidden steps or even perceive the order in which things occurred.

January of 1993 was spent cleaning and organizing previous work into a format that could be deciphered by someone other than the analyst. Instead of describing which files were moved where and how files were deleted, this section will summarize the file structures used in the course of this project.

Currently, work on the 1992-1993 cycle is being performed in

`paros:/users/clbs/ttemp/newwork`

while

`grover`

is where the post September 1, 1992 work for tree models is performed. The files on `paros` are either charts or post-September 1 work on the charts.

Filesystems involved:

`paros:/users/clbs/ttemp`

grover:/home8/clbs/dale
grover:/home8/clbs/ra/poly2
sparta:/home/clbs/ra

The directories on these file systems, and their contents are:

- paros:/users/clbs/ttemp/dmgt/
  This contains files used in processing charts in June of 1992

- paros:/users/clbs/ttemp/one-three/ and paros:/users/clbs/ttemp/five-twelve/
  These directories contain .Data directories with first year charts in S-PLUS format. The .Data directories also contain the Question Marker files and two lists of the charts in the nfilesxx and gfsxx files (xx refers to the number of the directory). The each of the main directories contains a file "questfile" containing the Question Text files, and a .zip file containing the raw .0x1 and .0x3 files.

- paros:/users/clbs/ttemp/work/:
  These files are mostly just the run/batch files, where the functions that sift through the databases to extract the features live.

- paros:/users/clbs/ttemp/work/csd/
  Cumulative spectral distribution work is in here

- paros:/users/clbs/ttemp/work/fdgfswork/
  Last run at frequency domain-September 10

- paros:/users/clbs/ttemp/work/functs/
  Description of functions and some old functions

- paros:/users/clbs/ttemp/work/gsrdswork/
  Runs of GSR frequency domain series-September 13

- paros:/users/clbs/ttemp/work/n2gfswork/
  Batch files used for creating n2gfs database (pff functions). These functions are mostly APL-like features.

- paros:/users/clbs/ttemp/work/vc/
  Harmonic regression work

- paros:/users/clbs/ttemp/work/.Data
  This contains a collection of functions called and plotting functions. It also contains four of the feature vector collections that were made after September 1. These are the ngfs, mmgfs, n2gfs and fdgfs files which correspond to some of the directories above.

- paros:/users/clbs/ttemp/newwork/:
  This directory contains work done in the second year of funding of the Polygraph Reliability project on the A and B datasets. Improved and object oriented functions can be found here.

- `grover:/home8/clbs/dale/fdgfsw/`
  Frequency domain work on Sept 14, 1992. V data 105 parameters

- `grover:/home8/clbs/dale/fullgfsw/`
  work Sept 17, 1992 on full dataset (276 X, Y and V parameters)

- `grover:/home8/clbs/dale/gsgfsw/`
  work on Sept 15; adds X data to V data

- `grover:/home8/clbs/dale/pdgfsw/`
  work Sept 11, V data.

- `grover:/home8/clbs/dale/rptwork/`
  Year One report with psfiles containing plots. Also contains files with plotting instructions.

- `grover:/home8/clbs/dale/tables/`
  Tables explaining what function or calculation was used to create a given variable. Includes some files back to early August. Data is summarized in tty.tables and in rept.tex in rptwork.

- `grover:/home8/clbs/ra/poly2`
  This is a directory where current work is being handled. The subdirectory monthly/ contains monthly reports and this summary. The reports.doug/ directory contains reports sent to Doug while he was in Taos. The mgqt/ directory has some initial work on the MGQT names and outcomes. [these have been copied to sparta:/home/clbs as of 6/11/93]

- `sparta:/home/clbs/ra/poly`
  These are older files used on the A and B datasets with APL parameters and which were mostly exercises in tree() modeling. The directory prog contains zipfiles of programs used to process Axciton files. (programs were revised in February 1993). The talk/ subdirectory contains files and plots used for APL talk by clbs on 6/17/1992. The talk concered tree based models and methodology.

- `sparta:/home/clbs/ra/rpts`
  This directory contains the workplans that were submitted to funding agencies in January of 1993. A rough timeline of the projected work is included in the files and separately.

- `sparta:/home/clbs/ra/oldst/wstd/`
  These directories contain data management files that were used in early Summer of 1992 at the beginning of my RA work with feature extraction. (This was before September 1.) We used question by question standardization which turned out to be inferior to later methods. Some of the datasets were deleted because they were just taking up space. They could be recalculated using the new feature extraction functions and amending them to standardize question by question (rather than by pairs or vs. strongest response).

- `sparta:/home/clbs/ra/oldstd/gfs.masterwork/`
  Contains a lot of old work-much of which is useless but saved anyway

- `sparta:/home/clbs/ra/oldstd/zzip/`
  Contains very early work (July) compressed using the zip format.

6 March-June 1993 work on MGQT charts

6.1 Summary

MGQT files were processed and the decryption routine was shared with other NSA/PERSEREC contractors. A procedure for adding new charts to the database was written.

6.2 Details

NSA notified us that other contractors would be needing some assistance on the polygraph project and that we should share our knowledge with them. Dr. John Angus from the Claremont Graduate School contacted us in late February about working with the polygraph data. In March, Dr. Ben Knapp at San Jose State University contacted us about reading charts as well. Both were supplied with instructions on how to use our decompression routines and are at the stage where they can process charts from compressed form into ASCII. The interaction with these researchers was lengthy, involving 87kb of correspondence.

NSA sent a set of MGQT data in March so that a classification tool could be devised for this type of chart. Along with 12 disks of charts, we also received copies of information on the cases and how they were scored by the examiners.

The MGQT format is considerably different from the zone test in that the question ordering is very different. The "standard" MGQT format is in table 1.

The DOS MGQT floppies were restored on a PC into a single file using the DOS 5.0 *restore* command. This file was transferred over to the UNIX machines and unzipped using the UNIX program `zip`, leaving us with 970 files containing 43,653,259 bytes of information. Because the file names contained reserved UNIX characters such as $, % and #, the reserved

Table 1: MGQT Question Order

1. Irrelevant	6. Control
2. Irrelevant	7. Irrelevant
3. Relevant	8. Relevant
4. Irrelevant	9. Relevant
5. Relevant	10. Control

characters in the files were deleted by renaming the files using the *mv* command. The files were then moved to directories according to the first character of the file name to make the data management easier. These directories were called: `four, five, six, seven, eight, nine, aset` and `bset` and were used throughout the analysis. Figure 1 summarizes the overall process.

The next step in file processing was to create the ASCII formatted or "processed" files using the *nf* program. (*nf* is based upon code that NSA gave us in March of 1992.) A slight modification of *nf* was needed because an inversion flag had changed causing the program to exit prematurely. The inversion flag is part of the header information in the *.0?1 file which tells the decompression algorithm to read the values as either "positive" or "negative" unsigned short (16 bit) integers. The program expected the inversion flag to be either 0 or 1, but instead it was 126, causing an unexpected exit from the routine.

The *nf* program first reads the subjectname.0(chartnumber)1 file which contains the following information regarding the chart:

1. Length of Chart in seconds units
2. Number of Channels
3. Sample rate
4. Number of Questions
5. Compression flags for each channel
6. Event markers and their locations

To process the charts, the UNIX script file *decc* and *nf* ware first moved into each directory (using a UNIX batch file *datamgt*). The *decc* file uses the UNIX foreach command to rename the target file for the `nf` program. The name consists of a 'z' followed by the subject name

	Event Marker File $$xxxxxx.0x1	Chart Data File $$xxxxxx.0x2	Question Text File $$xxxxxx.0x3
Axciton File	-contains the length of the chart, the number of channels, and the position of the Event Markers in units of 1/30th of a second	-contains the digitized series values, formatted according to flags in the Event Marker File	-contains the script of questionss or a shorthand script of questions.
Processing Notes	Becomes the 5th column of ASCII file 0 = start asking a question 1 = done asking a question 2 = subject response to question 9 = No Event Marker (column is mostly 9's)	Becomes 1st-4th columns of ASCII file Column 1-GSR Column 2-Cardio Column 3-Upper Resp Column 4-Lower Resp	Files are pooled into UNIX files by directory. Later, these files will be examined for deviations from the standard ordering of questions

UNIX File

ASCII File Format (with column labels)

File Row	GSR	Cardio	UR	LR	EvMark
1	1983	1931	1482	1083	9
2	1983	1922	1483	1084	9
3	1983	1913	1483	1084	9
4	1983	1906	1483	1085	9
5	1983	1902	1483	1085	9

This ASCII file is read into S-PLUS to become the zxxxxxx.x chart data file, according to the labeling from the original files. The 5th column labeled EvMark comes from the 0's in the **Event Marker** file. A 0 marks the beginning of a new question. The position of the 0's in the file is summarized in the **Question Marker** file in S-PLUS, where row names are assigned to match the order in the question text file. For example in the **Question Marker** file below, 1681 indicates that in row 1681 or 1681/30=56 seconds into the exam, the examiner began to ask a Sacrifice Relevant question.

S-PLUS File

Chart Data File
zxxxxxx.x

1	1983	1931	1482	1083	9
2	1983	1922	1483	1084	9
3	1983	1913	1483	1084	9
4	1983	1906	1483	1084	9
5	1983	1902	1483	1085	9

Question Marker File
zxxxxxx.xq

TB	361	960
N	961	1680
SR	1681	2280
S	2281	2910
C1	2911	3510

Figure 1: File Types

with '0(chartnumber)' as the suffix. *decc* then invokes *nf* with subject.0(chartnumber)1 and the targetfile as arguments. The targetfile contains the digitized values from the Axciton in four columns, with a fifth column containing the event markers (0,1,2) separated by 9's.

After decrypting the files, the original charts are compressed with another batch file *datamgt2* which also sets up `.Data` subdirectories in each of the chart directories. This compression batch step is followed by an S-PLUS batch step *datamgt3* which reads in the charts to S-PLUS and compresses the ASCII charts using the commands in the file *readbat*. In addition to reading in the charts, the S-PLUS batch job also creates question marker files (subject.?q) in S-PLUS. Question marker files list the times of the beginning and end of questions and are used throughout the extraction process.

The above steps were performed on sparta and consumed over 250Mb of disk space.

The final step in processing the raw files is to extract the question files. This is done by copying all the *.0?3 files into one file so that they may be examined later for deviations from the standard ordering of questions. After this is done, the original datafiles can be removed to tape for storage.

The database tools allow for easy access to each chart. To do this, we create a main database directory called `paros:/users/clbs/mgqt/work` and us .First() functions in S-Plus to automatically link the main database directory `.Data` to the directories containing the processed charts `four, five, ... , bset`. Chart label names contained in files `nf4` , `nf5` ,..., are combined as `nf` in the `work` directory. Next, the chart description forms sent from NSA are examined and we record the examiner's judgement regarding deception for each subject.

To insure that the questions were associated with the proper labels by the automatic question allocation routines, we needed to verify the ordering of the questions. There are two ways that this is done. The simplest way is to examine the question text files and match the questions that the examiner recorded to the labels given by the automatic routines. A second verification step requires looking at the event marker files for each subject and checking to see that a "2" was pressed during the times indicated by the labels. For example, the first automatic label is "TB" for "The test is about to begin." If a 2 was pressed during the time

indicated by the label, then an error has occured since there should be no response to that statement. These two steps using the question texts and the event markers also help identify any stim tests that were sent to us.

A list of exams that were not MGQT tests or for which it was not possible to identify the questions was obtained. These exams are excluded from the feature building. Of 302 charts, 264 are usable leaving 88 of the 96 original subjects. A log of these deletions was kept on `paros:/users/clbs/mgqt/work/changelog.questions` and was forwarded to San Jose and Claremont.

The last step before calculating features for the MGQT charts was a brief rewriting of the caller() and the new.subj.call() functions so that the different format of questions would be properly standardized. After a few tests, the functions were found to be acceptable and feature extraction could begin.

7 March-June 1993 work on Zone methods

7.1 Summary

Improvements on the Zone test feature extraction algorithms were made concurrently with the MGQT work. Two of the goals in the Second Year proposal were to improve the method used to detrend the series and to more accurately estimate the fundamental frequency for the cardio channel. The Integrated Spectral Density has been added to the set of features that may be calculated using caller().

7.2 Details

In the first year, we used supersmoother, supsmu() in S-PLUS, to remove trend from the data. Supersmoother is a variable span smoother that smooths different segments of a given series with possibly different smoothing spans. A section that is smooth to begin with would have a wide span, while a section that is more variable would get a narrower span. The primary advantage to using supersmoother is its speed; supersmoother is much faster than our other smoothing choices.

This adaptive bandwidth used by supersmoother meant that we could not fix the smoothing span, making it difficult to predict the results of applying supersmoother to a new series. We also noticed that supersmoother was oversmoothing some of the respiratory series, taking out parts of the trend that we wanted preserved.

We produced several plots using different smmothing span's (found in `paros:/users/clbs/ttemp/n` and visual inspection lead us to believe that a fixed span of 0.6 was most appropriate for smoothing. A smaller span removed too much of the cyclic component from the respiratory channels, and a larger span does not filter out enough trend in the cardio channel.

A major benefit of the spectral analysis is exact estimation of the fundamental cardio frequency. See the First Year Report for details.

After changing the smoother, we saw more "pollution" of the lower frequencies of the spectrum—more of the low frequency trend from the respiratory channels would be left in the series since the bandwidth of the smoother was relatively high. In addition, we perceived that we were accepting fundamental cardio frequencies which were too low. We expected that these frequencies were actually respiratory frequencies, and that the interaction of respiration with heart volume was leading to our estimating cardio fundamental frequencies which were too low (since they were actually respiratory frequencies).

To see how well (or poorly) we could estimate the fundamental cardio frequency, the following experiment was tried: First, we estimated the cardio fundmental frequency using lower bounds for the cardio channels. Then we estimated these fundamental frequencies, but we did not restrict fundamental frequencies to be greater than our lower bounds. For the cardio channel, we saw that all cardio fundamental frequencies were above our lower bound. Without the bound some of our estimates for the fundamental cardio frequency fell below this bound and were actually respiratory frequencies.

Specifically, we declare the fundamental for cardio series as the greatest peak at a frequency about 0.0295. For a respiratory channel, we declare the greatest peak about 0.00325 as our fundamental.

These two methodological improvements did not translate into significant reduction in error rates for the Zone tests. We believe that the poor quality of many of the A and B

charts has limited our ability to improve our rates. Initial examination of the MGQT charts seems to indicate that most of these charts are free from artifacts. This knowledge, together with the careful screening of the questions on the MGQT charts, should give us a good basis for scoring this new dataset.

Appendix: Functions from December

```
"caller"<-
function(object, ...)
UseMethod("caller")

"caller.uni"<-
function(FUN, chartname, col = 2, ...)
{
#MGQT version
        qnm <- paste(chartname, "q", sep = "")  #
# get the question marker file
        qv <- get(qnm)  #
# how many questions are there?
        ddim <- dim(qv)[1]
        cvec <- rep(0, ddim)    #
# what are the labels for the questions?
        dn <- dimnames(qv)[[1]] #
# which were the Control questions
        for(i in 1:ddim) {
                if(dn[i] == "C1" | dn[i] == "C2")
                        cvec[i] <- 1
        }
        rvec <- rep(0, ddim)    #
# which were the relevant questions
        for(i in 1:ddim) {
                if(dn[i] == "R1" | dn[i] == "R2" | dn[i] == "R3"
                       | dn[i] == "R4") rvec[i] <- 1
        }
        pvec <- cvec + rvec     #
#  How many relevants
#(this is different from Zone!!!)
        psum <- sum(rvec)
        pcount <- c(1:ddim)
        pc <- pcount[pvec == 1]
        rc <- pcount[rvec == 1]
```

```
        cc <- pcount[cvec == 1] #
# first two Rel's get paired with 1st control, 2nd two with 2nd control
        cc <- c(rep(cc[1], 2), rep(cc[2], 2))
        xo <- get(chartname)[, col]
        dn2 <- dn[c(1:(psum * 2))]      #
# this stores the question labels
        for(i in 1:psum) {
                dn2[(2 * i - 1)] <- dn[cc[i]]
                dn2[(2 * i)] <- dn[rc[i]]
        }
#
# storage for the results...
# actually, this could be done with
# retur_eval(FUN,(xo,c(cc[1],rc[1]),qv.col,...))
# retur_rbind(retur,eval(FUN,(xo,c(cc[i],.....
        retur <- matrix(0, psum * 2, attr(FUN, "nargs"))
        for(i in 1:psum) {
                retur[(2 * i - 1):(2 * i),  ] <- eval(FUN(xo,
                        c(cc[i], rc[i]), qv, col, ...))
        }
        return(data.frame(length = retur, lab = dn2))
}

caller.biv<-
function(FUN, chartname, col = c(2, 3), ...)
{
        qnm <- paste(chartname, "q", sep = "")
        qv <- get(qnm)
        ddim <- dim(qv)[1]
        cvec <- rep(0, ddim)
        dn <- dimnames(qv)[[1]]
        for(i in 1:ddim) {
                if(dn[i] == "C1" | dn[i] == "C2")
                        cvec[i] <- 1
        }
        rvec <- rep(0, ddim)
        for(i in 1:ddim) {
                if(dn[i] == "R1" | dn[i] == "R2" | dn[i] == "R3" | dn[i] ==
                        "R4")
                        rvec[i] <- 1
        }
        pvec <- cvec + rvec
        psum <- sum(rvec)
        pcount <- c(1:ddim)
```

```
        pc <- pcount[pvec == 1]
        rc <- pcount[rvec == 1]
        cc <- pcount[cvec == 1]
        cc <- c(rep(cc[1], 2), rep(cc[2], 2))
        xo <- get(chartname)[, col]
        dn2 <- dn[c(1:(psum * 2))]
        for(i in 1:psum) {
                dn2[(2 * i - 1)] <- dn[cc[i]]
                dn2[(2 * i)] <- dn[rc[i]]
        }
        retur <- matrix(0, psum * 2, attr(FUN, "nargs"))
        for(i in 1:psum) {
           retur[(2 * i - 1):(2 * i),  ] <- eval(FUN(xo, ccc[i],
                      rc[i]), qv, col, ...))
        }
        return(data.frame(length = retur, lab = dn2))
}

call.gs1 <- function(datai, quests, qvm, ...)
{
        cl <- quests[1]
        rl <- quests[2] #
# Right here is where the real work will come in
#
        data <- mst(c(datai[qvm[cl, 1]:(qvm[cl, 2])],
                 datai[qvm[rl, 1]:(qvm[rl,2])]))
        cdata <- data[c(1:(qvm[cl, 2] - qvm[cl, 1]))]
        tempc <- diff(cdata, 1, 1)
        tem1 <- median(tempc[1:60])
        tem2 <- median(tempc[1:150])
        tem3 <- median(tempc[1:240])
        tem4 <- mean(tempc[1:(length(tempc))])
        tem5 <- max(cdata) - min(cdata)
        tem6 <- var(cdata)
        rdata <- data[ - c(1:(qvm[cl, 2] - qvm[cl, 1]))]
        tempr <- diff(rdata, 1, 1)
        tem21 <- median(tempr[1:60])
        tem22 <- median(tempr[1:150])
        tem23 <- median(tempr[1:240])
        tem24 <- mean(tempr[1:(length(tempr))])
        tem25 <- max(rdata) - min(rdata)
        tem26 <- var(rdata)
        return(rbind(c(tem1, tem2, tem3, tem4, tem5, tem6), c(tem21, tem22,
                tem23, tem24, tem25, tem26)))
```

```
}
attr(, "class"):
[1] "uni"
attr(, "nargs"):
[1] 6

gnct <- function(treet, df, df.tr)
{
#  treet is a tree, df is a data.frame and df.tr is the associated
#  set of true outcomes for the df
#  This is called after finding the best tree of charts (not subjects)
#  gnct(fnd1.t1,nd1,nd1.tr)
#  calls nclass()
        pre1 <- predict.tree(treet, df)[, 2]
        pre.df <- data.frame(df.tr, t(nclass(pre1, df)))
        names(pre.df)[1] <- c("out")
        return(pre.df)
}

nclass <- function(fit, df)
{
#  This function combines data from the chart level to the subject level
#  INPUTS:  fit--fitted values from gnct call --fits from chart tree
#           df --df used to create chart tree
#  OUTPUTS: returns matrix with median, mean and extreme value
#           calculation for each subject.

        person <- unique(df$short)
        k <- length(person)
        fitmean <- function(j, fit, lev.las, las2)
        {
                dat <- fit[las2 == lev.las[j]]
                tempm <- median(dat)
                tempm2 <- mean(dat)
                tempm3 <- ifelse(abs(1 - max(dat)) < abs(min(dat)),max(dat),
                        min(dat))
                return(c(tempm, tempm2, tempm3))
        }
        men <- matrix(0, k, 3)
        men <- apply(as.matrix(c(1:k)), 1, fitmean, fit, person,df$short)
        dimnames(men) <- list(c("median", "mean", "ext"), person)
        return(men)
}
```

```
gnch <- function(predv, subs, ds, ds1a)
{
#INPUTS:   predv=predicted values from subject tree,
#          subs=number to check which subset is worked on 3=whole data
#                       1=a data,0=bdata
#          ds=dataset used to grow subject tre
#          ds1a=indicator of a's in subject dataset
#gnch(predict(tm1f,newdata=nd1.ct)[,2],3,nd1.ct,nd1.ad)
#gnch(predict(tm1f,newdata=nd1.ct[nd1.ad==1,])[,2],1,nd1.ct,nd1.ad)
#gnch(predict(tm1f,newdata=nd1.ct[nd1.ad==0,])[,2],0,nd1.ct,nd1.ad)
#output 3 ratios of error rates in full,a and b data sets.

        ddim <- length(predv)
        res.out <- predv * 0
        res.out <- round(predv, 0)
        if(subs < 2) {
             res.v2 <- sum(abs(as.numeric(ds[ds1a == subs, 1]) - 1- res.out))
        }
        else {
                res.v2 <- sum(abs(as.numeric(ds[, 1]) - 1 - res.out))
        }
        ret <- paste(res.v2, "/", length(predv), "  = ",
                round(res.v2/length(predv), 3))
        return(ret)
}
"new.subj.call"<-
function(chartname, colnum, FUN, ...)
{
#MGQT version
        nargs <- attr(FUN, "nargs")     #
# Here I start off by cheating--I need to know how many
# charts to expect from a subject, so I look it up in master.list
#
        chl <- length(lch <- master.list$chartnames[master.list$short ==
                chartname])     #
#
# Set up storage for each of the features
        for(k in 1:nargs) {
                assign(paste("ressld", k,  , sep = ""), c(0, 0))
        }
#
# get the right storage mode for resslc
        resslc <- c("1")
```

```
        cht <- 0        #
# loop over the number of charts from the same subject
        for(i in 1:chl) {
                chartname <- lch[i]     #
# calculate feature using Feature extraction function FUN
                resul <- eval(caller(FUN, chartname, colnum, ...))      #
# for the first one, we just set things up
#
# this next part is just a set of gymnastics in Splus to get it
# to allocate space in a "dynamic" manner
# the ressld{1,,nargs} are just vectors that paste more
# stuff into, the assign() call might be cleaned up, but I do not
# see a simple way to do it....
# perhaps you could just stack a matrix together by column or rows....
# (I designed this for a different version of S and this was all that
# I could get to work...)
                if(i == 1) {
                     for(k in 1:nargs) {
                         assign(paste("ressld", k,  , sep = ""), resul[,k])
                     }
                     resslc <- as.character(resul$lab)
                     cht <- rep(i, length(resslc))
                }
                else {
                     for(k in 1:nargs) {
                        assign(paste("ressld", k,  , sep = ""), c(get(
                            paste("ressld", k,  , sep = "")), resul[, k]))
                        }
                        resslc <- c(resslc, as.character(resul$lab))
                        cht <- c(cht, rep(i, length(resul$length.1)))
                }
        }
#       okay, now standardize
        resn <- rep(0, length(resslc))  #
# set up dummy variables to code questions
        for(i in 1:length(resslc)) {
                resn[i] <- resn[i] + ifelse(resslc[i] == "C1", 1, 0)
                resn[i] <- resn[i] + ifelse(resslc[i] == "C2", 1, 0)
                resn[i] <- resn[i] + ifelse(resslc[i] == "R1", 3, 0)
                resn[i] <- resn[i] + ifelse(resslc[i] == "R2", 4, 0)
                resn[i] <- resn[i] + ifelse(resslc[i] == "R3", 5, 0)
                resn[i] <- resn[i] + ifelse(resslc[i] == "R4", 6, 0)
        }
#
```

```
# matrix of 4 differences
        outt <- matrix(0, chl, nargs * 4)
        ucht <- unique(cht)
        ncht <- length(ucht)    #
#
#
# This is one way to "standardize" the subject scores, you can
# probably think of many others--
# if you just reported the scores for each question (say 6
# questions for each chart/for each feature) they you could manipulate
# the scores directly and figure out another way of "standardizing" without
# having to recalculate features
        for(i in 1:ncht) {
          for(k in 1:nargs) {
             workk <- get(paste("ressld", k, sep = ""))
             diffs <- rep(0, 4)
             nquest.cht <- length(cht[cht == ucht[i] & resn > 2])
             for(l in 1:nquest.cht) {
                avg <- mean(workk[resn < 2 & cht == ucht[i]])
                diffs[l] <- workk[resn == (l + 2) & cht == ucht[i]] - avg
             }
             if(nquest.cht < 3)
                diffs[3] <- 0
             if(nquest.cht < 4)
                diffs[4] <- 0
             outt[i, (4 * k - 3)] <- diffs[1]
             outt[i, (4 * k - 2)] <- diffs[2]
             outt[i, (4 * k - 1)] <- diffs[3]
             outt[i, (4 * k)] <- diffs[4]
          }
        }
        return(outt)
}
```

5 Appendix III

The following gives a list of the charts removed from the MGQT data with a brief description of the reason these charts were removed. The names used here may be slightly different from the names used in the Axciton files because of UNIX naming conventions.

1. z51de36.3 - remove from listing-stim, no event markers
2. z80d.3q - throw out, cannot tell what questions are
3. z84jhc.2 - stim, remove
4. z8djgl.1,2,3 - ZONE, remove from MGQT listings
5. z8qybjo.1 - remove, no idea what questions are
6. z8zsqcr.1,2,3,4 - cannot tell what the questions are–remove
7. z92wqwk.1 - stim, remove
8. z9iooxo.1,.3 - stim,remove
9. z9llic3.1,.2,.3 - ZONE, remove from MGQT listings
10. z9su7.1,.2 - stim,remove
11. z9w0b9f.2 - stim,remove
12. za6uif.2 - stim, do not use
13. zarfjf.1,.2,.3 - remove, cannot tell what the questions are
14. zawa8j.2 - stim, do not use
15. zbcma7p.3 - remove-not enough questions (no C2)
16. zbk3ff.1,.2,.3,.4 - ZONE, remove from MGQT listings
17. zawa8j.1 - examiner called the charts Inconclusive
18. zawa8j.2 - examiner called the charts Inconclusive
19. zawa8j.3 - examiner called the charts Inconclusive
20. z9szc.1 - examiner called the charts Inconclusive:
21. z9szc.2 - examiner called the charts Inconclusive:
22. z9szc.3 - examiner called the charts Inconclusive:
23. z9sqdf9.1 - examiner called the charts Inconclusive:

24. z9sqdf9.2 - examiner called the charts Inconclusive:
25. z9sqdf9.3 - examiner called the charts Inconclusive:
26. zc8egp0.1 - wrong test (REMOVE)
27. zc8egp0.2 - wrong test (REMOVE)
28. zc8egp0.3 - wrong test (REMOVE)
29. zcdpkp0.2 - stim (REMOVE)
30. zciuz1u.2 - stim, flat GSR, unusable (REMOVE)
31. zcm5y56.2 - flat GSR, stim test, (REMOVE)
32. z7oxm60.1 - order of questions not determined (REMOVE)
33. z7oxm60.2 - order of questions not determined (REMOVE)
34. z7oxm60.3 - order of questions not determined (REMOVE)
35. z7r51p9.1 - order of questions not determined (REMOVE)
36. z7r51p9.2 - movement, order of questions not determined (REMOVE)
37. z7r51p9.3 - order of questions not determined (REMOVE)
38. z7rh0ro.1 - order of questions not determined (REMOVE)
39. z7rh0ro.2 - order of questions not determined (REMOVE)
40. z7rh0ro.3 - order of questions not determined (REMOVE)
41. z80d.2 - movement in "I", bad question order (REMOVE)
42. z8fgmvi.1 - strange GSR - bad event markers, order is tb, i1, i2, r1, i3, r2, c1, i4, i4, r3, r4, c2 (REMOVE)
43. z8nb6u.1 - questions (REMOVE)
44. z8nb6u.2 - questions (REMOVE)
45. z8nb6u.3 - questions (REMOVE)
46. z8qmts.1 - strange GSR (REMOVE)
47. z8qmts.3 - order is impossible to determine (REMOVE)
48. z8ys26.1 - No questions (REMOVE)
49. z8ys26.2 - No questions (REMOVE)

50. z8ys26.3 - No questions (REMOVE)

51. z92wqwk.2 - No questions (REMOVE)

52. z92wqwk.3 - No questions (REMOVE)

53. z92wqwk.4 - movement in R2, no questions (REMOVE)

54. z9iooxo.5 - order is tb, i1, i2, r4, c1, i3, r1, c2, r2, c1, r3, r3, c2, i4, question R3 asked twice, questions hard to determine (REMOVE)

55. za3hirx.3 - order is unknown (REMOVE)

56. zbq4shi.1 - already in data - two 2's in a row (REMOVE)

57. zbq4shi.2 - already in data - spike in audio 3 indicates remove (REMOVE)

58. zbq4shi.3 - already in data - order is TB, I4, I1, R2, C1, R1, C2, R4, C1, R3, C2, XX, spikes (REMOVE)

59. zbr6R60.1 - already in data - spikes in I questions - order is TB, i1, i2, R1, i3, C1, i4, i5, R3, R4, C2, XX (REMOVE)

60. zbr6R60.2 - already in data - very low GSR in I1 (REMOVE)

61. zbr6R60.3 - already in data - order is TB, i4, i1, R4, C1, i2, R1, C2, R2, i4, C1, R3, C2, XX (REMOVE)

62. zbtu99c.1 - already in data - spike in i1 (REMOVE)

63. zbtu99c.2 - already in data - stim (REMOVE)

64. zbtu99c.3 - already in data (REMOVE)

65. zbtu99c.4 - already in data - order is TB, i4, i1, r4, c1, i2, R3, c2, r2, c1, r3, r4, xx (REMOVE)

66. zcDygsl.1 - order is tb, i1, i2, c1, r1, i3, r2, c2, r3, i4, r4, c3, xx - too many controls (REMOVE)

67. zcDygsl.2 - order is tb, i1, i2, c1, r1, i3, r2, c2, r3, i4, r4, c3, xx - too many controls - (REMOVE)

68. zcDygsl.3 - order is tb, i4, i5, c3, r4, i3, r1, c1, r3, i4, r2, c2, xx - too many controls - (REMOVE)

69. zcbf5z6.1 - spike in i1 - not enough questions (REMOVE)

70. zcg6j24.3 - order is tb, i4, i1, i2, c1, i2, r1, c2, r2, c1, r3, c2, xx - no r4 (REMOVE)

71. zcjlv4r.1 - order is tb, i1, i2, r1, c1, r2, c2, r3, c3, i4, xx too man controls (REMOVE)

72. zcjlv4r.2 - order is tb, i1, i2, r1, c1, r2, c2, r3, c3, i4, xx too many controls (REMOVE)

73. zcjlv4r.3 - order is tb, i1, i2, r1, c1, r2, c2, r3, c3, i4, xx too many controls (REMOVE)

74. zcjlv4r.4 - order is tb, i1, i2, r1, c1, r2, c2, r3, c3, i4, xx too many controls (REMOVE)

75. zcr3aye.2 - not enough questions (REMOVE)

76. zcre6zy.2 - bad event markers - (REMOVE)

77. zcre6zy.4 - bad event markers - (REMOVE)

78. zctccdR.3 - bad event markers - (REMOVE)

79. zcwqzju.1 - order is tb, i1, i2, c1, r1, i3, r2, c2, r3, i4, r4, c3, i5, xx - too many controls (REMOVE)

80. zcwqzju.2 - order is tb, i1, i2, c1, r1, i3, r2, c2, r3, i4, r4, c3, i5, xx - too many controls (REMOVE)

81. zcwqzju.3 - chart not complete (REMOVE)

82. zcwqzju.4 - order is wrong - too many controls (REMOVE)

83. zcyl5kk.3 - bad event markers - (REMOVE)

84. zcyx3sP.1 - No question file - (REMOVE)

85. zcyx3sP.3 - No question file - (REMOVE)

86. zd185v6.2 - too few questions - (REMOVE)

87. zd4x9vo.1 - too many controls - (REMOVE)

88. zd4x9vo.2 - too many controls - (REMOVE)

89. zd4x9vo.3 - too many controls - (REMOVE)

90. zd5Rdf4.2 - too few questions - (REMOVE)

91. zd6riRo.2 - bad GSR - too few questions - (REMOVE)

92. zd6riRo.5 - too few questions - (REMOVE)

93. zd6riRo.6 - too few questions - (REMOVE)

94. zdb879c.1 - jump in r2 cardio - order is tb, i1, i2, c1, r1, i3, r2, c2, i4, xx - not enough relevants - (REMOVE)

95. zdb879c.2 - wrong order - not enough relevants - (REMOVE)

96. zdb879c.3 - wrong order - not enough relevants - (REMOVE)

97. zdbzpoR.2 - bad event markers - (REMOVE)

98. zdbzpoR.4 - bad event markers - (REMOVE)

6 Appendix IV

Listed here with a brief description is a list of the feature sets and features used in this study. Also given is a listing of the channels each feature set was computed on. These channels are: 1 – galvanic skin response, 2 – cardiograph, 3 – upper pneumograph, and 4 – lower pneumograph.

- GS1 - Channels 1, 2, and 4.

    1. median60 – median of the first 60 lagged values;
    2. median150 – median of the first 150 lagged values;
    3. median240 – median of the first 240 lagged values;
    4. mean – mean of the standardized data;
    5. range – range of the standardized data; and
    6. variance – variance of the standardized data.

- G1 – Channels 1, 2, and 4.

    1. med240 – the median of the first 240 observations;
    2. med241 – the median of the differences from 241 to the minimum of 420 or the end of the series;
    3. sp25 – the 25-th percentile for the smoothed, differenced data;
    4. sp75 – the 75-th percentile for the smoothed differenced data;
    5. MAXAMP – the amplitude of the maximum spectrum value greater than the lowest allowed frequency in the standardized data minus its smooth; and
    6. MAXFREQ – the frequency at which the greatest amplitude occurs.

- G2 – Channels 1, 2, and 4.

    1. minimum – the minimum of the standardized data;
    2. maximum – the maximum of the standardized data;
    3. midrange – the midrange (0.75 - 0.25) of the elements from the 60-th to the minimum of the 420-th or the end of the questions data;
    4. mid.9.1 – the midrange (0.9 - 0.1) of the elements from the 60-th to the minimum of the 420-th or the end of the questions data;
    5. mid.8.2 – the midrange ((0.8-0.2) of the first 240 data points; and
    6. midi.8.2 – the midrange (0.8-0.2) of the first 120 data points.

- G2A – Channels 1, 2, and 4, smoothed data.

    1. minimum – the minimum of the standardized data;

2. maximum – the maximum of the standardized data;
3. midrange – the midrange (0.75 - 0.25) of the elements from the 60-th to the minimum of the 420-th or the end of the questions data;
4. mid.9.1 – the midrange (0.9 - 0.1) of the elements from the 60-th to the minimum of the 420-th or the end of the questions data;
5. mid.8.2 – the midrange ((0.8-0.2) of the first 240 data points; and
6. midi.8.2 – the midrange (0.8-0.2) of the first 120 data points.

- G3 – Channels (2,4) together.

1. phase1 – the phase of the first channel at the fundamental frequency
2. coher1 – the coherency of the first channel at its fundamental frequency;
3. phase2 – the phase of the second channel at the fundamental frequency;
4. coher2 – the coherency of the second channel at the fundamental frequency;
5. phaser1 – the phase of the first channel at the fundamental frequency, reduced serie;
6. coherr1 – the coherency of the first channel at its fundamental frequency, reduced series;
7. phaser2 – the phase of the second channel at the fundamental frequency, reduced series; and,
8. coherr2 – the coherency of the second channel at the fundamental frequency, reduced series.

The reduced series is the first 240 observations (8 seconds) after the beginning of the question.

- G4 – Channels 2 and 4.

1. fundfreq – the fundamental frequency;

- G5 – Channels 1, 2, and 4.

1. mxabsm – The maximum absolute difference between the control and the relevant cumulative spectrums, standardized data;
2. smabsm – the sum of the absolute differences between the control and relevant cumulative spectrums, standardized data;
3. smsqsm – the sum of the squared differences between the control and relevant cumulative spectrums, standardized data;
4. mxabrw – The maximum absolute difference between the control and the relevant cumulative spectrums, smooth-removed data;

5. smabrw – the sum of the absolute differences between the control and relevant cumulative spectrums, smooth-removed data; and
6. smsqrm – the sum of the squared differences between the control and relevant cumulative spectrums, smooth-removed data.

- FD2 – Channels (2, 4) together.

1. fund1 – the fundamental frequency, first channel;
2. fund2 – the fundamental frequency, second channel;
3. amp1 – the amplitude of the spectrum for channel 1 at the fundamental frequency for channel 2;
4. amp2a1 – the amplitude of the spectrum for channel 2 for the fundamental frequency for channel 1;
5. amp1h1 – the amplitude of the spectrum for channel 1 at the first harmonic for channel 1;
6. amp1h2 – the amplitude of the spectrum for channel 1 at the second harmonic for channel 1;
7. amp1h3 – the amplitude of the spectrum for channel 1 at the third harmonic for channel 1;
8. amp2a1 – the amplitude of the spectrum for channel 2 for the fundamental frequency for channel 2;
9. amp2h1 – the amplitude of the spectrum for channel 2 at the first harmonic for channel 2;
10. amp2h2 – the amplitude of the spectrum for channel 2 at the second harmonic for channel 2;
11. amp2h3 – the amplitude of the spectrum for channel 2 at the third harmonic for channel 2;
12. coh2 – the coherency of the spectrum at the fundamental frequency for channel 2;
13. coh1 – the coherency of the spectrum at the fundamental frequency for channel 1;
14. coh1h1 – the coherency of the spectrum at the first harmonic for channel 1;
15. coh1h2 – the coherency of the spectrum at the second harmonic for channel 1
16. coh1h3 – the coherency of the spectrum at the third harmonic for channel 1;

7 Appendix V

The following pages contain the help files for many of the S-Plus functions written for the polygraph project at the University of Washington.

axcitron.conv	Converts an ASCII Axcitron File to an S File	**axcitron.conv**

DESCRIPTION

The axcitron polygraph data comes in binary files which must be interpreted to produce usable data. For each Axcitron chart, routine axcitron.conv() produces, in the working directory, an S-Plus chart matrix, and a question ordering file used by later routines.

USAGE

axcitron.conv(chartname)

REQUIRED ARGUMENTS

chartname The axcitron name of the file. This is base.0c1, where base is the six character base name, and c is the chart number.

VALUE

Returns nothing.

SIDE EFFECTS

Creates two S-Plus file in frame 1. The file zbase.c contains the chart data, and the file zbase.cq contains the question ordering data (where the questions begin, and what they contain. If the question ordering file is has less than four questions, a message is printed.

EXAMPLES

axcitron.conv("clwjbf.031")

bxplt Function to provide comparative boxplots for D and ND groups **bxplt**

DESCRIPTION

For each outcome measure in a dataframe, bxplt() produces a plot with the boxplots of the observations for the D versus the ND groups.

USAGE

bxplt(df, outcome)

REQUIRED ARGUMENTS

df A dataframe containing the outcome measures to be plotted.

outcome A vector with length equal to the length of the data frame containing the classification of each observation.

VALUE

No values are returned.

EXAMPLES

bxplt(df, outcome)

call.fd2 Statistics Related to the Fundamental Frequency **call.fd2**

DESCRIPTION

Computes the fundamental frequency and its harmonics/amplitudes for each channel.

USAGE

call.fd2(data, quests, qmarker, channel, ...)

REQUIRED ARGUMENTS

data The vector of channel data.
quests The questions in data for which features are to be extracted.
qmarker The marker file for this chart.
channel The channel that represented by data.

OPTIONAL ARGUMENTS

... Optional argument to the spec function, used to calculate the spectral analysis.

VALUE

A 2 by 16 array containing the extracted features. Row one contains features for the control question (always a vector of zeros), and row two contains features for the relevant question. The data is first standardized by concatenating the control and relevant data together and then subtracting the median and dividing by the mean absolute deviation. Next supsmu is used with a span of 0.6 in order to obtain a smooth for the data. Statistics are computed on the standardized data, and then on the smooth-removed standardized data. Statistics computed are: 1 - the fundamental frequency, first channel; 2 - the fundamental frequency, second channel; 3 - the amplitude of the spectrum for channel 1 at the fundamental frequency for channel 2; 4 - the amplitude of the spectrum for channel 2 for the fundamental frequency for channel 1; 5 - the amplitude of the spectrum for channel 1 at the first harmonic for channel 1; 6 - the amplitude of the spectrum for channel 1 at the second harmonic for channel 1; 7 - the amplitude of the spectrum for channel 1 at the third harmonic for channel 1; 8 - the amplitude of the spectrum for channel 2 for the fundamental frequency for channel 2; 9 - the amplitude of the spectrum for channel 2 at the first harmonic for channel 2; 10 - the amplitude of the spectrum for channel 2 at the second harmonic for channel 2; 11 - the amplitude of the spectrum for channel 2 at the third harmonic for channel 2; 12 - the coherency of the spectrum at the fundamental frequency for channel 2; 13 - the coherency of the spectrum at the fundamental frequency for channel 1; 14 - the coherency of the spectrum at the first harmonic for channel 1; 15 - the coherency of the spectrum at the second harmonic for channel 1; 16 - the coherency of the spectrum at the third harmonic for channel 1;

EXAMPLES

Use caller (or caller.uni) to call call.fd2.

call.g1 Simple Statistics on Smoothed Data **call.g1**

DESCRIPTION

Computes some simple statistics on the smoothed differenced data.

USAGE

call.g1(data, quests, qmarker, channel, ...)

REQUIRED ARGUMENTS

data The vector of channel data.
quests The questions in data for which features are to be extracted.
qmarker The marker file for this chart.
channel The channel that represented by data.

OPTIONAL ARGUMENTS

... Optional argument to the spec function, used to calculate the spectral analysis.

VALUE

A 2 by 6 array containing the extracted features. Row one contains features for the control question, and row two contains features for the relevant question. The data is first standardized by concatenating the control and relevant data together and then subtracting the median and dividing by the mean absolute deviation. Next supsmu is used with a span of 0.6 in order to obtain a smooth for the data. The smooth is then differenced, and statistics are computed on the differenced data. Statistics computed are: 1 - the median of the first 240 observations; 2 - the median of the differences from 241 to the minimum of 420 or the end of the series; 3 - the 25-th percentile for the smoothed, differenced data; 4 - the 75-th percentile for the smoothed differenced data; 5 - the amplitude of the maximum spectrum value greater than the lowest allowed frequency in the standardized data minus its smooth; and 6 - the frequency at which the greatest amplitude occurs.

EXAMPLES

Use caller (or caller.uni) to call call.g1.

call.g2 Simple Channel Statistics on Standardized Data **call.g2**

DESCRIPTION

Computes some simple range statistics on standardized, and on differenced, data.

USAGE

call.g2(data, quests, qmarker, channel, ...)

REQUIRED ARGUMENTS

data A vector of channel data.
quests The questions in data for which features are to be extracted.
qmarker The marker file for this chart.
channel The channel that represented by data.

OPTIONAL ARGUMENTS

... Not used.

VALUE

A 2 by 6 array containing the extracted features. Row one contains features for the control question, and row two contains features for the relevant question. The data is first standardized by concatenating the control and relevant data together and then subtracting the median and dividing by the mean absolute deviation. For some features, a lag 1 difference is also computed. Statistics computed are: 1 - the minimum of the standardized data; 2 - the maximum of the standardized data; 3 - the midrange (0.75 - 0.25) of the elements from the 60-th to the minimum of the 420-th or the end of the questions data; 4 - the midrange (0.9 - 0.1) of the elements from the 60-th to the minimum of the 420-th or the end of the questions data; 5 - the midrange ((0.8-0.2) of the first 240 data points; and 6 - the midrange (0.8-0.2) of the first 120 data points.

EXAMPLES

Use caller (or caller.uni) to call call.g2.

call.g3 Phase statistics for two channels **call.g3**

DESCRIPTION

Computes spectral phase statistics for two channels in the data.

USAGE

call.g3(data, quests, qmarker, channels, ...)

REQUIRED ARGUMENTS

data A nrow by two matrix of channel data.
quests The questions in data for which features are to be extracted.
qmarker The marker file for this chart.
channels The channels that represented by data.

OPTIONAL ARGUMENTS

... Optional arguments for the spec routine.

VALUE

A 2 by 8 array containing the extracted features. Row one contains features for the control question, and row two contains features for the relevant question. The data is first standardized by concatenating the control and relevant data together and then subtracting the median and dividing by the mean absolute deviation. Next supsmu is used with a span of 0.6 in order to obtain a smooth for the data. Statistics are computed on the difference of the standardized and the smoothed data. Computed statistics are: 1 - the phase of the first channel at the fundamental frequency; 2 - the coherency of the first channel at its fundamental frequency; 3 - the phase of the second channel at the fundamental frequency; 4 - the coherency of the second channel at the fundamental frequency; 5 - the phase of the first channel at the fundamental frequency, reduced serie; 6 - the coherency of the first channel at its fundamental frequency, reduced series; 7 - the phase of the second channel at the fundamental frequency, reduced series; and, 8 - the coherency of the second channel at the fundamental frequency, reduced series. The reduced series is the first 240 observations (4 second) after the beginning of the question.

EXAMPLES

Use caller (or caller.uni) to call call.gs3.

call.g4 Computes the Fundamental Frequency **call.g4**

DESCRIPTION

Given a channel of data, this feature extraction function computes the fundamental frequency.

USAGE

call.g4(data, quests, qmarker, channel, ...)

REQUIRED ARGUMENTS

move the above line to just above the first optional argument

data The vector of channel data.

quests The questions in data for which features are to be extracted.

qmarker The marker file for this chart.

channel The channel that represented by data.

OPTIONAL ARGUMENTS

... Optional arguments for the spec() function.

VALUE

A 2 by 2 array containing the extracted features. Row one contains features for the control question, and row two contains features for the relevant question. The data is first standardized by concatenating the control and relevant data together and then subtracting the median and dividing by the mean absolute deviation. Next supsmu is used with a span of 0.6 in order to obtain a smooth for the data. The smooth is then differenced, and statistics are computed on the differenced data. Statistics computed are: 1 - the fundamental frequency; and 2 - 0. For the relevant question, this becomes: 1 - 0; and 2 - fundamental frequency.

SIDE EFFECTS

EXAMPLES

Use caller (or caller.uni) to call call.g4.

call.g5	Kullback-Liebler Distance Features	call.g5

DESCRIPTION

Computes difference measures based upon the spectrums of the control and the relevant questions.

USAGE

call.g5(data, quests, qmarker, channel, ...)

REQUIRED ARGUMENTS

data The vector of channel data.
quests The questions in data for which features are to be extracted.
qmarker The marker file for this chart.
channel The channel that represented by data.

OPTIONAL ARGUMENTS

... Optional argument to the spec function, used to calculate the spectral analysis.

VALUE

A 2 by 6 array containing the extracted features. Row one contains features for the control question (always a vector of zeros), and row two contains features for the relevant question. The data is first standardized by concatenating the control and relevant data together and then subtracting the median and dividing by the mean absolute deviation. Next supsmu is used with a span of 0.6 in order to obtain a smooth for the data. Statistics are computed on the standardized data, and then on the smooth-removed standardized data. Statistics computed are: 1 - The maximum absolute difference between the control and the relevant cumulative spectrums, standardized data; 2 - the sum of the absolute differences between the control and relevant cumulative spectrums, standardized data; 3 - the sum of the squared differences between the control and relevant cumulative spectrums, standardized data; 4 - The maximum absolute difference between the control and the relevant cumulative spectrums, smooth-removed data; 5 - the sum of the absolute differences between the control and relevant cumulative spectrums, smooth-removed data; and 6 - the sum of the squared differences between the control and relevant cumulative spectrums, smooth-removed data.

EXAMPLES

Use caller (or caller.uni) to call call.g5.

call.gs1 Simple Descriptive Features **call.gs1**

DESCRIPTION

Simple descriptive features for lag 1, standardized data.

USAGE

call.gs1(data, quests, qmarker, ...)

REQUIRED ARGUMENTS

~move the above line to just above the first optional argument

data The data for the channel for which features are to be extracted.

quests Two questions, a control, and the relevant, for which features are desired.

qmarker The marker data, obtained from the "q" file.

OPTIONAL ARGUMENTS

... Not yet used.

VALUE

A two by six array containing the extracted features. Row one contains the data from the control, and row two contains the data for the relevant question. The data is first standardized by concatenating data from the two questions. The resulting vector is standardized by subrtacting the median and dividing by the mean absolute deviation, and then a lag 1 difference is taken. Statistic computed are: 1 - median of the first 60 lagged values; 2 - median of the first 150 lagged values; 3 - median of the first 240 lagged values; 4 - mean of the standardized data; 5 - range of the standardized data; and 6 - variance of the standardized data.

EXAMPLES

Use caller (or caller.uni) to call call.gs1.

caller	Calls the appropriate feature extraction function.	caller

DESCRIPTION

This function is generic (see Methods); method functions can be written to handle specific classes of data. Classes which already have methods for this function in- clude: uni, biv

USAGE

caller(FUN, ...)

REQUIRED ARGUMENTS

FUN The feature extraction function.

OPTIONAL ARGUMENTS

... methods may have additional arguments.

VALUE

Varies from method to method.

SEE ALSO

caller.uni, caller.biv, caller.pl2

EXAMPLES

caller(call.gs1, master.list[1,1])

caller.biv Calls Feature Extraction Functions for a Two Channels **caller.biv**

DESCRIPTION

Calls a single channel feature extraction function with data for two channels.

USAGE

caller.biv(FUN, chartname, channels=(2, 3), ...)

REQUIRED ARGUMENTS

FUN The feature extraction function.

chartname A character string giving the name of the chart for which features are to be computed.

OPTIONAL ARGUMENTS

channels A vector giving the two channels for which features are to be computed. Channels are 1 - galvanic skin response; 2 - cardio; 3 - upper respiratory; and 4 - lower respiratory. Exactly two channels at a time must be specified.

... Optional arguments passed to the feature extraction function.

VALUE

A data.frame with rows equal to twice the number of relevant questions and columns equal to the number of features the feature extraction function computes. The rows alternate between control and then a paired relevant question.

SEE ALSO

new.subj.call

EXAMPLES

caller.biv(call.gs1, master.list[1,1])

caller.pl2	Displays the Polygraph Charts	caller.pl2

DESCRIPTION

USAGE

caller.pl2(FUN, chartname, channels=c(1, 2, 4), ...)

REQUIRED ARGUMENTS

FUN The feature extraction function.

chartname A character string giving the name of the chart for which features are to be computed.

channels A vector giving the channels to be displayed. These are 1 - galvanic skin response; 2 - cardio; 3 - upper respiratory; and 4 - lower respiratory.

OPTIONAL ARGUMENTS

... Not currently used.

VALUE

zero.

SIDE EFFECTS

Displays a plot of the channels.

EXAMPLES

caller.pl2(pl3, chartname)

caller.uni	Calls Feature Extraction Functions for a Single Channel	caller.uni

DESCRIPTION

Calls a single channel feature extraction function.

USAGE

caller.uni(FUN, chartname, channel=2, ...)

REQUIRED ARGUMENTS

FUN The feature extraction function.

chartname A character string giving the name of the chart for which features are to be computed.

OPTIONAL ARGUMENTS

channel A scalar giving the channel for which features are to be computed. These are 1 - galvanic skin response; 2 - cardio; 3 - upper respiratory; and 4 - lower respiratory. Only one channel at a time may be specified.

... Optional arguments passed to the feature extraction function.

VALUE

A data.frame with rows equal to twice the number of relevant questions and columns equal to the number of features the feature extraction function computes. The rows alternate between control and then a paired relevant question.

SEE ALSO

new.subj.call

EXAMPLES

caller.uni(call.gs1, master.list[1,1])

chartplot	Plots the Polygraph Channels on the Active Device	**chartplot**

DESCRIPTION

Plots the polygraph channels on the active graphical device.

USAGE

chartplot(chartname, cols=c(1, 2, 4), rows=c(0), qs=c(0), ...)

REQUIRED ARGUMENTS

chartname A character string containing the name of the char to plot.

OPTIONAL ARGUMENTS

cols The polygraph channels to plot.

rows An interger vector of length two containing the row numbers of the rows of polygraph data to plot. If a single zero is entered, all rows are plotted.

qs The question numbers in the poluygraph chart to plot. Currently this does not seem to work correctly.

... Plotting parameters to be passed to the high level plotting routines.

VALUE

Nothing is returned.

SIDE EFFECTS

A single plot is produced.

EXAMPLES

chartplot("zcr0g9i.1")

check.chart	Checks the PolyGraph Chart Questions and plots	**check.chart**

DESCRIPTION

Displays three channels on a polygraph chart for a single individual on an existing graphical device, and then uses check.questions to display the questions (which must be available in a file base.0c3q, where base is the 6 character base name with no prepended z, and c is the chart number.

USAGE

check.chart(chartname)

REQUIRED ARGUMENTS

chartname The name of the chart to check. This has format zbase.c, where base is the six character base name, and c is the chart number.

VALUE

The questions and other chart information are displayed in the S-Plus command window, and a plot of the chart channels is produced. No value is returned.

SEE ALSO

check.questions

EXAMPLES

```
check.chart("zc9xmo.1")
```

check.labels	Checks chart labels	**check.labels**

DESCRIPTION

Checks chart question labels to make sure that each of the four relevent questions is asked only once, that each of the two control questions is asked at least once, and that the other question labels are known.

USAGE

check.labels(chartname)

REQUIRED ARGUMENTS

chartname A character string giving the chart name.

VALUE

Nothing is returned. Bad charts are listed.

EXAMPLES

```
check.labels(as.character(master.list[20,1]))
```

check.questions	Displays the Questions used in a Polygraph Chart	**check.questions**

DESCRIPTION

Displays the questions used in a polygraph chart, and gives the question ordering.

USAGE

check.questions(chartname)

REQUIRED ARGUMENTS

chartname A character string giving the chart name using format zbase.c, where base is the base chart name, and c is the chart number.

VALUE

A list containing the following components is returned.

questions A data.frame contains the question type, and the question. Unfortunately, question types vary from chart to chart.

codes A vector of codes in the

dnames A vector of character strings assigned to each question. These are hard coded, and thus the actual questions asked can be quite different from the characterization of question types given in dnames. Codes used in dnames are: "TB" - the test is about to begin; "i1", "i2", ... - irrelevent question 1, 2, ...; "C1", "C2", ... - control question 1, 2, ...; "R1", "R2", ... - relevent question 1, 2, ...

DETAILS

Function check.questions() is used to display the questions for a polygraph chart. It assumes that the working directory contains these questions in a file with name base.0c3q, where base is the 6 character base name with no prepended z, and c is the chart number. The main use of check.questions() is in the function check.chart(), which also gives a graphical display of the polygraph chart.

SEE ALSO

check.chart

EXAMPLES

check.chart("zc9xmo.1")

compute.features	Computes a feature vector for a single chart	**compute.features**

DESCRIPTION

Returns a labeled row vector containing features for a single chart.

USAGE

compute.features(chartname, channels, FUN, ...)

REQUIRED ARGUMENTS

chartname A character string containing the name of the chart for which features are desired.

channel A channel (or channels) for which features are desired. Only one channel (or set of two channels) at a time may be specified.

FUN The feature extraction function.

OPTIONAL ARGUMENTS

... Arguments used by the feature extraction function.

VALUE

A labelled row vector containing the extracted features.

SEE ALSO

new.subj.call

EXAMPLES

compute.features(master.list[1,1], c(2), call.gs1)

do.tree	Function to Analyze Features	**do.tree**

DESCRIPTION

Creates a tree, fits it, cross-validates error rates, chooses the best tree, and then adds selected variables to a data.frame().

USAGE

do.tree(x, merge=T, print.it=T, add.data=NULL, COMPARE=get.compare, subset=NULL)

REQUIRED ARGUMENTS

~move the above line to just above the first optional argument

x Matrix containing the features for which an analysis is desired.

OPTIONAL ARGUMENTS

merge If merge is true, a subject wise analysis is performed. Otherwise, the analysis is by charts.

print.it If true, printing is performed.

add.data If not null, data is added to the indicated data.frame.

COMPARE The function to use when comparing the relevant and control questions.

subset a logical vector giving the subset of data to use for this analysis. The remaining data is used in estimating the probabilities of misclassification.

VALUE

Nothing is returned.

SIDE EFFECTS

A files with the analysis results is created and printed, and a plot of the results is made (but only if print.it=T). If add.data is not null, the selected features are added to data.frame add.data.

DETAILS

SEE ALSO

do.summary.tree

EXAMPLES

do.tree(g1.df)

mst	Functions to Standardize a Vector	mst

DESCRIPTION

Used to standardize a channel using robust methods.

USAGE

mst(x)

REQUIRED ARGUMENTS

x The vector to standardize.

VALUE

A standardized vector.

DETAILS

Standardization consists of subtracting the median and dividing by the mean absolute deviation.

EXAMPLES

mst(x)

post.chart	Postscript Plots of Polygraph Charts	post.chart

DESCRIPTION

Produces a postscript plot of at most two polygraph charts.

USAGE

post.chart(chartname, chartname2=NULL)

REQUIRED ARGUMENTS

chartname A character string giving the name of the first polygraph chart to plot.

OPTIONAL ARGUMENTS

chartname2 If given, a character string giving the name of the second polygraph chart to plot.

VALUE

No value is returned.

SIDE EFFECTS

Produces a plot on the postscript device. Invokes, and then turns off, the postscript driver.

SEE ALSO

chartplot

EXAMPLES

post.chart("zav53p6.3" "zawa38x.1")

questm	Finds Polygraph Examiner Codes	**questm**

DESCRIPTION

For a polygraph chart, examines the code channel, and determines which channels are different from 9. Returns a matrix containing the chart locations of zero codes.

USAGE

questm(chart)

REQUIRED ARGUMENTS

chart The chart object for which the channel code matrix is desired.

VALUE

A two column matrix containing the locations of the beginning and ending time marks for each question asked.

DETAILS

The polygraph examiner marks each question event with an event marker. These are: 0 - start asking the question; 1 - finish asking the question; and 2 - the subject responds to the question. A nine indicates that there was no event. The routine questm() finds the beginning and ending event markers for each question in the polygraph test.

EXAMPLES

questm("z51qep3.1")

spec	Estimate Spectrum	spec

DESCRIPTION

Estimates the spectrum of a time series.

USAGE

spec(x, method="pgram", plot=T, spans=1, pad=0, taper=0.1, detrend=T, demean=F, n.freq=n2, frequency=1

REQUIRED ARGUMENTS

x a univariate or multivariate time series, or a vector, or a matrix with a univariate series per column. Missing values are allowed only at the ends.

OPTIONAL ARGUMENTS

method If method="pgram", a periodogram is used with spec.pgram. If method="ar", estimates based upon autoregression are used.

plot If plot=T, a plot of the spectral density is produced.

spans a sequence of lengths of modified Daniell smoothers to run over the raw periodogram. Use spans = 1, the default, for the raw periodogram. A modified Daniell smoother has all values equal except for the 2 end values which are half the size of the others. The values should be odd integers.

pad fraction of the length of x that is to be padded: {pad*length(x)} zeros are added to the end of the series before computing the periodogram.

taper fraction of each end of the time series that is to be tapered. A split cosine taper is applied to {taper*length(x)} points at each end of series. This must take values between 0 and 0.5.

detrend if TRUE, remove a least squares line from each component of the series before computing periodogram.

demean if TRUE, remove the mean of each series before computing the periodogram (detrend also removes the mean).

n.freq the number of frequencies between 0 and the Nyquist frequency (=frequency/2 cycles per unit time) at which to compute the spectrum. The default value is n.used/2+1, where n.used is the number of observations not missing in the original time series. This must be supplied if n.used is not a component of ar.list.

frequency the sampling frequency for the time series. The default is the frequency of the resid component of ar.list if present, and 1 otherwise.

VALUE

a list containing the following components:

freq the vector of frequencies between 0 and the Nyquist at which the spectrum estimate is computed.

spec a vector if a univariate series, and otherwise a matrix with columns representing univariate series and rows corresponding to the frequencies in freq. The spectrum estimate is in decibels (10 times log to base 10 transformation).

coh a matrix containing the squared coherencies between each pair of series. If j is less than k, then the pair (j,k) corresponds to column (k-1)(k-2)/2 + j. For univariate series this component is NULL.

phase a matrix like coh containing the phase differences between each pair of series (only for multivariate time series). The units are radians. These are made continuous by requiring that the first differences be less than pi. To put them back into the range [0, 2 pi) use the modulus operator %%: e.g. reducedphase <- phase%%(2*pi). For univariate series this component is NULL.

order	the order of the autoregression in ar.list.
method	a string describing the method used.
series	a string containing the name of the time series, if available from ar.list.

SIDE EFFECTS

If plot=TRUE, a plot of the spectrum is produced.

EXAMPLES

Called by the feature extraction functions.q